IFW

PTO/SB/21 (09-04)
Approved for use through 07/31/2006. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Total Number of Pages in This Submission	23

Application Number	10/065,324
Filing Date	October 3, 2002
First Named Inventor	Mats HOLGERSSON
Art Unit	3721
Examiner Name	Gloria Weeks
Attorney Docket Number	03485.0004.NPUS00

ENCLOSURES *(Check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☐ Amendment/Reply
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Response to Missing Parts/ Incomplete Application
  - ☐ Response to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☒ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s) _____
  - ☐ Landscape Table on CD
- ☐ After Allowance Communication to TC
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to TC **(Appeal Notice, Brief, Reply Brief)**
- ☐ Proprietary Information
- ☐ Status Letter
- ☒ Other Enclosure(s) (please Identify below): Exhibits A-C and Postcard

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name	Howrey LLP		
Signature			
Printed name	Michael J. Bell		
Date	March 9, 2006	Reg. No.	39,604

CERTIFICATE OF TRANSMISSION/MAILING

I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below.

Signature			
Typed or printed name		Date	

This collection of information is required by 37 CFR 1.5. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and1.14. This collection is estimated to 2 hours to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Confirmation No. 9313
HOLGERSSON, Mats	Art Unit: 3721
Appl. No. 10/065,324	Examiner: WEEKS, Gloria
Filed: October 3, 2002	Atty. Docket: 03485.0004.NPUS00
For: CONTROL DEVICE FOR A DRIVE MOTOR IN A STAPLER	

Renewed Petition under 37 C.F.R. § 1.181

Mail Stop **Petition**
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

This Renewed Petition is being submitted in response to the Decision on Petition dated March 7, 2006 (attached as Exhibit A and incorporated herein by reference). In the aforementioned Decision on Petition, the Petitions Attorney indicated that the abandonment of the application by the USPTO was improper. (Exhibit A at 1.) However, the fee for an extension of time was not able to be charged because the person that signed the Deposit Account authorization (Tracey Druce) is no longer an authorized user of the referenced deposit account. (Exhibit A at 2.) Therefore, the Applicant submits this Renewed Petition with the proper Deposit Account authorization.

I. ***Request to Withdraw Holding of Abandonment***

As stated in the Applicant's previous petition, the Applicant submitted a Reply to the outstanding Office Action (a copy of which is attached hereto as Exhibit B) on June 17, 2004 in the above-referenced application. However, the Applicant received a Notice of Abandonment (a copy of which is attached hereto as Exhibit C) on August 26, 2004, alleging that the Reply was deemed improper for lack of fees for extension of time.

With respect to the grounds cited in support of the issuance of the Notice of Abandonment, the Office's attention is respectfully directed to the final page of the Reply (Exhibit B at 12) where Applicant states the following:

The undersigned representative requests any extension of time that may be deemed necessary to further the prosecution of this application.

The undersigned representative authorizes the Commissioner to charge any additional fees under 37 C.F.R. 1.16 or 1.17 that may be required, or credit any overpayment, to Deposit Account No. 08-3038, referencing Order No. 03485.0004.NPUS00.

It is respectfully asserted that this statement constitutes a proper request for an Extension of Time according to M.P.E.P. § 710.02(e) in that the request was timely made (in this case, explicitly), and authorization was provided for deposit account payment. Therefore, it is respectfully asserted that the Notice of Abandonment was incorrectly issued and should be withdrawn.

II. ***Deposit Account Authorization***

The Commissioner is hereby authorized by the undersigned to charge the fee for a three-month extension of time and any other chargers that might be due to Deposit Account No. 08-3038, referencing attorney docket number 03485.0004.NPUS00.

Further, it is not believed that extensions of time or fees beyond those that may otherwise be provided for in documents accompanying this paper. However, if additional extensions of time are necessary to prevent abandonment of this application, then such extensions of time are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required therefor (including fees for net addition of claims) are hereby authorized to be charged to our Deposit Account No. 08-3038 referencing docket number 03485.0004.NPUS00.

Respectfully submitted,

Michael J. Bell (Reg. No. 39,604)

Date: March 9, 2006

HOWREY LLP
2941 Fairview Park Drive, Box 7
Falls Church, VA 22042
(703) 663-3600

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

HOWREY LLP
C/O IP DOCKETING DEPARTMENT
2941 FAIRVIEW PARK DR
SUITE 200
FALLS CHURCH VA 22042-2924

COPY MAILED

MAR 0 7 2006

OFFICE OF PETITIONS

In re Application of
Mats Holgersson :
Application No. 10/065,324 : DECISION ON
Filed: October 3, 2002 : PETITION
Attorney Docket No. 03485.0004.NP :

This is in response to the "PETITION TO WITHDRAW HOLDING OF ABANONMENT OR ALTERNATIVELY PETITION TO REVIVE UNINTENTIONAL ABANDONMENT", filed September 8, 2004. The petition was recently forwarded to the Office of Petitions for consideration.

The petition under 37 CFR 1.181 is **DISMISSED**.

Any request for reconsideration of this decision must be submitted within **TWO (2) MONTHS** from the mail date of this decision. The reconsideration request should include a cover letter entitled "Renewed Petition under 37 CFR 1.181". Extensions of time under 37 CFR 1.136(a) are permitted. No fee is required for a renewed petition.

The above-identified application was held abandoned for failure to timely file a reply to the non-final Office action mailed December 17, 2003. No extensions of time under 37 CFR 1.136(a) were obtained. No response having been received, the above-identified application became abandoned on March 18, 2004. A Notice of Abandonment was mailed on August 26, 2004.

A review of the application file reveals that applicant did file a response on June 17, 2004. Contained within the text of the response was an authorization to charge the fees for any

necessary extension of time to Deposit Account No. 08-3038.[1] A review of Office records indicates that the deposit account contained a sufficient balance on June 17, 2004 to charge $950 for a three month extension of time. The fee was not charged.

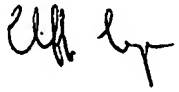
However, Office records indicate that the person signing the Deposit Account authorization on June 17, 2004, Tracey Druce, is no longer an authorized user on Deposit Account No. 08-3038. Accordingly, the fee for a three month extension of time can not be charged at this time. On renewed petition, petitioner must present a Deposit Account authorization from an authorized user.

Further correspondence with respect to this matter should be addressed as follows:

By mail: Mail Stop Petitions
Commissioner for Patents
P.O. Box 1450
Alexandria VA 22313-1450

By FAX: (571) 273-8300
Attn: Office of Petitions

Telephone inquiries related to this decision should be directed to the undersigned at 571-272-3207.



Cliff Congo
Petitions Attorney
Office of Petitions

---

[1] For future reference, petitioner is reminded of 37 CFR 1.4(c), which states that "since different matters may be considered by different branches or sections of the United States Patent and Trademark Office, each distinct subject, inquiry, or order must be contained in a separate paper to avoid confusion and delay in answering papers dealing with different subjects."

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: HOLGERSSON, Mats	Group Art Unit: 3721
Serial No.: 10/065,324	Examiner: WEEKS, Gloria R.
Confirmation No.: 9313	Atty. Dkt. No.: 03485.0004.NPUS00
Filed: 10/03/2002	
For: CONTROL DEVICE FOR A DRIVE MOTOR IN A STAPLER	

CERTIFICATE of TRANSMISSION

I hereby certify that this correspondence and any attachments referred to herein are being facsimile transmitted to Examiner WEEKS, Art Unit 3721, United States Patent and Trademark Office at fax no. **703.872.9306** on **17 June 2004**.

Daniel Hernandez

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: HOLGERSSON, Mats	Group Art Unit: 3721
Serial No.: 10/065,324	Examiner: WEEKS, Gloria R.
Confirmation No.: 9313	Atty. Dkt. No.: 03485.0004.NPUS00
Filed: 10/03/2002	
For: CONTROL DEVICE FOR A DRIVE MOTOR IN A STAPLER	

Request for Reconsideration in Response to Non-Final Office Action

---

INTRODUCTORY COMMENTS

The following is being provided in response to the Non-Final Office Action mailed December 17, 2003. The Applicant respectfully requests reconsideration based on the following remarks. The Applicant respectfully submits that the claims as presented are in condition for allowance.

AMENDMENTS TO THE CLAIMS

Please amend the claims as follows:

1. (Withdrawn) A control device (21) for controlling an electric stapler (1), said control device comprising:

a microprocessor (22), an electrical drive motor (2) that is incorporated in a stapler (1) and whose drive shaft (9) drives a staple driver (13) in a forward and reverse motion that has a defined start point and a defined reversing point, and which staple driver drives, during its forward motion, a staple (15) into a workpiece (17); and

said control device (21) further comprising a sensor (23) that senses the rotational speed of the drive shaft (9) and the degree of rotation it has completed from the start point, the control device then transfers the sensed information to the microprocessor (22) that analyzes the obtained information and generates a control signal that controls the supply of current to the drive motor (2) whereupon the rotational speed of the drive shaft is regulated.

2. (Withdrawn) The control device according to claim 1, wherein said workpiece is a sheaf of paper.

3. (Withdrawn) The control device according to claim 1, wherein the supply of current occurs across a full bridge (27), whereupon the supply of current is controlled so that the rotational direction and speed of the drive shaft (9) is regulated.

4. (Previously Presented) A method for controlling an electric stapler, said method comprising:

arranging a sensor for detecting positions of a staple driver of the electric stapler during execution of a stapling process;

utilizing a microprocessor to analyze sensed position information about the staple driver; and

controlling positional changes of the staple driver based on the analysis of sensed position information.

5. (Original) The method for controlling an electric stapler according to claim 4, further comprising:

affecting positional changes of the staple driver by supplying controlled magnitudes of electrical current to a drive motor that is interconnected with the staple driver.

6. (Original) The method for controlling an electric stapler according to claim 4, further comprising:

affecting directional changes in movement of the staple driver by supplying opposite sense electrical current to a drive motor interconnect with the staple driver.

7. (Original) The method for controlling an electric stapler according to claim 6, further comprising:

utilizing a circuit internal to the electric stapler for switching the sense of the electrical current that is supplied to the drive motor that is interconnect with the staple driver.

8. (Original) The method for controlling an electric stapler according to claim 4, further comprising:

detecting positions of a staple driver on a real time basis.

9. (Original) The method for controlling an electric stapler according to claim 8, further comprising:

computing speed of travel of the staple driver based on a series of positions of the staple driver taken on a real time basis.

10. (Previously Amended) The method for controlling an electric stapler according to claim 4, further comprising:

driving said staple driver via an electric motor having a drive shaft rotatingly extending therefrom; and

utilizing said sensor to detect and report rotational speed of the drive shaft.

11. (Original) The method for controlling an electric stapler according to claim 10, further comprising:

utilizing said sensor to detect and report rotational positions of the drive shaft.

12. (Original) The method for controlling an electric stapler according to claim 10, further comprising:

interconnecting the drive shaft with the staple driver via at least one toothed gear thereby establishing a directly proportional relationship between rotational characteristics of the drive shaft and translational characteristics of the staple driver.

13. (Original) The method for controlling an electric stapler according to claim 10, further comprising:

interconnecting the drive shaft with the staple driver via a plurality of interacting toothed gears thereby establishing a directly proportional relationship between rotational characteristics of the drive shaft and translational characteristics of the staple driver.

14. (Original) The method for controlling an electric stapler according to claim 10, further comprising:

reversing the direction of travel of the staple driver based on a sensed degree of rotation of the drive shaft correlating to a completed stapling action.

15. (Original) The method for controlling an electric stapler according to claim 10, further comprising:

slowing the travel speed of the staple driver upon approach to a completion position of a stapling action.

16. (Original) The method for controlling an electric stapler according to claim 4, further comprising:

reversing the direction of travel of the staple driver when a completion position of a stapling action is sensed and prior to a detrimental load being imposed upon the staple driver and an associated powering transmission.

17. (Original) The method for controlling an electric stapler according to claim 16, further comprising:

arranging the associated powering transmission to include at least a drive shaft extending from a driving electrical motor, a plurality of interacting toothed gears and an interacting toothed rack.

18. (Currently Amended) The method for controlling an electric stapler according to claim 17, further comprising:

arranging a the toothed rack to be a part of the staple driver.

19. (Withdrawn) The method for controlling an electric stapler according to claim 1, further comprising utilizing a control device to control the electric stapler, said control device comprising:

a microprocessor, an electrical drive motor and a drive shaft configured to drive a staple driver in a forward and reverse motion that defines a start point and a reversing point, the staple driver configured to drive, during forward motion, a staple into a workpiece; and

the control device further comprising a sensor adapted to sense a rotational speed of the drive shaft and a degree of rotation relative to the start point, the control device being further adapted to transfer sensed information to the microprocessor that analyzes the obtained information and generates a control signal that controls the supply of current to the drive motor whereby the rotational speed of the drive shaft is regulated.

20. (Withdrawn) The method for controlling an electric stapler according to claim 19, wherein said workpiece is a sheaf of paper.

21. (Withdrawn) The method for controlling an electric stapler according to claim 19, wherein the supply of current occurs across a full bridge whereby the supply of current is controlled so that the rotational direction and speed of the drive shaft is regulated.

22. (New) A method for controlling an electric stapler, said method comprising:

providing a drive shaft driven electric stapler for coupling together a multi-member workpiece;

sensing, during a stapling movement, drive shaft speed and degree of rotation of the drive shaft measured from a start point; and

adjusting the electric supply of current that powers the drive shaft during the stapling process in dependence upon the sensed drive shaft speed and measured degree of rotation.

23. (New) The method for controlling an electric stapler according to claim 22, wherein the supply of current is increased when high resistance to drive shaft movement is detected.

24. (New) The method for controlling an electric stapler according to claim 22, wherein the supply of current is decreased upon approach to a reverse position of the electric stapler.

REMARKS

In response to the Office Action dated December 17, 2003, the Applicant respectfully requests reconsideration based on the following remarks. The Applicant respectfully submits that the claims as presented are in condition for allowance.

The United States Patent and Trademark Office (the "Office") restricted this application to claims 4-18. The Office has since rejected claim 18 under 35 U.S.C. § 112, paragraph two, as being indefinite. The Office also rejected claims 4-18 under 35 U.S.C. §102 (b) as being anticipated by U.S. Patent 5,230,457 to Hiroi *et al.* The Applicant shows, however, that the pending claims are patentably distinguishable over *Hiroi*, and the Applicant thus respectively submits that the pending claims are ready for allowance.

Rejection Under § 102 (b)

The Office rejected claims 4-18 under 35 U.S.C. §102 (b) as being anticipated by U.S. Patent 5,230,457 to Hiroi *et al.* In response, the Applicant requests that the Examiner reconsider and withdraw the rejection in view of the following remarks.

A claim is anticipated only if "each and every element" of the claimed invention is found either expressly or inherently described in a single prior art reference. *See Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 U.S.P.Q. 2d (BNA) 1051, 1053 (Fed. Cir. 1987). *See also Kloster Speedsteel AB v. Crucible Inc.*, 793 F.2d 1565, 1571, 230 U.S.P.Q. (BNA) 81, 84 (Fed. Cir. 1986) ("absence from the reference of any claimed element negates anticipation."); *In re Schreiber,* 128 F.3d 1473, 1477, 44 U.S.P.Q.2d (BNA) 1429, 1431 (Fed. Cir. 1997). As pointed out by the court, "[t]he identical invention must be shown in as complete detail as is contained in the ... claim." *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236, 9 U.S.P.Q.2d (BNA) 1913, 1920 (Fed. Cir. 1989). An anticipating reference must describe the patented subject matter with sufficient clarity and detail to establish that the subject matter

existed and that its existence was recognized by persons of ordinary skill in the field of the invention. *See ATD Crop. V. Lydall, Inc.*, 159 F.3d 534, 545, 48 U.S.P.Q. 2d (BNA) 1321, 1328 (Fed. Cir. 1998). *See also In re Spada*, 911 F.2d 705, 708, 15 U.S.P.Q. 2d (BNA) 1655, 1657 (Fed. Cir. 1990). *See also* DEPARTMENT OF COMMERCE, MANUAL OF PATENT EXAMINING PROCEDURE, § 2131 (orig. 8th Edition) (hereinafter "M.P.E.P."). As the Applicant shows, however, independent claim 4, and thus the dependent claims thereunder, are patentably distinguishable over *Hiroi*. The reference to *Hiroi* does not anticipate this invention, so the Applicant respectfully requests that Examiner Weeks remove the 35 U.S.C. § 102 (b) rejection of claims 4-18.

1. *Hiroi* Does NOT Control Position of the Stapler Based on "Sensed Position Information"

Hiroi does not anticipate independent claim 4. Claim 4 requires "controlling positional changes of the staple driver *based on the analysis of sensed position information*" (emphasis added). *Hiroi*, in contradistinction to this invention, analyzes current magnitudes when controlling the position of the stapler. If the current amperage is above a certain threshold, a "jam" is assumed and the motor is reversed. If the current amperage is low, a "blank shot" condition is assumed. As *Hiroi* explains, a current sensor is used to infer staple loads (*see* column 3, lines 66-69). If the current is within a "normal" range, the stapling operation is "deemed normal" (*Hiroi* at column 5, lines 3-5). If the current is higher than expected, it is assumed that a jam or other malfunction has occurred (*Hiroi* at column 5, lines 5-7). If the current is lower than expected, it is assumed that a "blank shot" has occurred (*Hiroi* at column 5, lines 7-9). A terminal of the controller "receives a detection signal from the current sensor" (*Hiroi* at column 5, lines 42-43). "The control circuit monitors the signal from the current sensor, that is, the current *I* through the motor" (*Hiroi* at column 5, lines 49-51). If the peak current is normal, "the home position of the stapler is confirmed, and the motor is stopped" (*Hiroi* at column 6, lines 1-7). If, however, "the current level is excessively high," the motor is reversed (*Hiroi* at column 6, lines 8-13).

Hiroi, then, does not anticipate independent claim 4. Claim 4 requires "controlling positional changes of the staple driver *based on the analysis of sensed position information*" (emphasis added). *Hiroi*, again in contradistinction to this invention, only analyzes current magnitudes controlling the position of the stapler. *Hiroi*, then, does not control position of the stapler based on analysis of "sensed position information." Because *Hiroi* measures current through the motor, *Hiroi* cannot anticipate this invention. The Applicant respectfully requests that Examiner Weeks remove the 35 U.S.C. § 102 (b) rejection of independent claim 4.

The dependent claims are also not anticipated. Because *Hiroi* does not anticipate independent claim 4, the dependent claims 5-18 are likewise unanticipated. The Applicant, then, respectfully asks Examiner Weeks to remove the 35 U.S.C. § 102 (b) rejection of claims 4-18.

For the above reasons the Applicant submits that nowhere in the *Hiroi* reference is there believed to be any disclosure or suggestion to modify the structure shown in a manner to achieve the claimed invention. In view of the above, the Applicant requests the reconsideration and withdrawal of the rejection of claims 4-18 under 35 U.S.C. § 102 (b). The Applicant also asks that the Examiner indicate the allowance of the claims in the next paper from the Office.

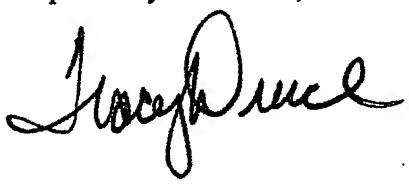
The new claims have been added to alternatively claim the presently elected invention. The features recited therein regarding sensing stapler drive shaft speed and degree of rotation are asserted as being patentable over any singular, or appropriate combination of the references of record.

The undersigned representative requests any extension of time that may be deemed necessary to further the prosecution of this application.

The undersigned representative authorizes the Commissioner to charge any additional fees under 37 C.F.R. 1.16 or 1.17 that may be required, or credit any overpayment, to Deposit Account No. 08-3038, referencing Order No. 03485.0004.NPUS00.

In order to facilitate the resolution of any issues or questions presented by this paper, the Examiner should directly contact the undersigned by phone to further the discussion.

Respectfully submitted,



Tracy W. Druce
Patent Attorney
Reg. No. 35,493

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/065,324	10/03/2002	Mats Holgersson	03485.0004NP	9313

22930 7590 08/26/2004

HOWREY SIMON ARNOLD & WHITE LLP
ATTEN: MARGARET P. DROSOS, DIRECTOR OF IP ADMIN
2941 FAIRVIEW PARK DR, BOX 7
FALLS CHURCH, VA 22042

EXAMINER	
WEEKS, GLORIA R	

ART UNIT	PAPER NUMBER
3721	

DATE MAILED: 08/26/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Notice of Abandonment	Application No.	Applicant(s)	
	10/065,324	HOLGERSSON, MATS	
	Examiner	Art Unit	
	Gloria R Weeks	3721	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

This application is abandoned in view of:

1. ☒ Applicant's failure to timely file a proper reply to the Office letter mailed on *17 December 2003*.
   (a) ☐ A reply was received on _____ (with a Certificate of Mailing or Transmission dated _____), which is after the expiration of the period for reply (including a total extension of time of _____ month(s)) which expired on _____.
   (b) ☐ A proposed reply was received on _____, but it does not constitute a proper reply under 37 CFR 1.113 (a) to the final rejection.
   (A proper reply under 37 CFR 1.113 to a final rejection consists only of: (1) a timely filed amendment which places the application in condition for allowance; (2) a timely filed Notice of Appeal (with appeal fee); or (3) a timely filed Request for Continued Examination (RCE) in compliance with 37 CFR 1.114).
   (c) ☒ A reply was received on *17 June 2004* but it does not constitute a proper reply, or a bona fide attempt at a proper reply, to the non-final rejection. See 37 CFR 1.85(a) and 1.111. (See explanation in box 7 below).
   (d) ☐ No reply has been received.

2. ☐ Applicant's failure to timely pay the required issue fee and publication fee, if applicable, within the statutory period of three months from the mailing date of the Notice of Allowance (PTOL-85).
   (a) ☐ The issue fee and publication fee, if applicable, was received on _____ (with a Certificate of Mailing or Transmission dated _____), which is after the expiration of the statutory period for payment of the issue fee (and publication fee) set in the Notice of Allowance (PTOL-85).
   (b) ☐ The submitted fee of $_____ is insufficient. A balance of $_____ is due.
   The issue fee required by 37 CFR 1.18 is $_____. The publication fee, if required by 37 CFR 1.18(d), is $_____.
   (c) ☐ The issue fee and publication fee, if applicable, has not been received.

3. ☐ Applicant's failure to timely file corrected drawings as required by, and within the three-month period set in, the Notice of Allowability (PTO-37).
   (a) ☐ Proposed corrected drawings were received on _____ (with a Certificate of Mailing or Transmission dated _____), which is after the expiration of the period for reply.
   (b) ☐ No corrected drawings have been received.

4. ☐ The letter of express abandonment which is signed by the attorney or agent of record, the assignee of the entire interest, or all of the applicants.

5. ☐ The letter of express abandonment which is signed by an attorney or agent (acting in a representative capacity under 37 CFR 1.34(a)) upon the filing of a continuing application.

6. ☐ The decision by the Board of Patent Appeals and Interference rendered on _____ and because the period for seeking court review of the decision has expired and there are no allowed claims.

7. ☒ The reason(s) below:

Applicant's response was deemed improper due to lack of fees for extension of time. An attempt was made to contact Applicant's representative, Tracy Druce (Reg. No. 35,493), however, Attorney Druce was unavailable for further correspondance.

SCOTT A. SMITH
PRIMARY EXAMINER

Petitions to revive under 37 CFR 1.137(a) or (b), or requests to withdraw the holding of abandonment under 37 CFR 1.181, should be promptly filed to minimize any negative effects on patent term.

O'REILLY
ONLamp.com
LAMP: THE OPEN SOURCE WEB PLATFORM

Published on ONLamp.com (http://www.onlamp.com/)
http://www.onlamp.com/pub/a/onlamp/2001/08/16/ldap.html
See this if you're having trouble printing code examples

An Introduction to LDAP

by Luke A. Kanies
08/16/2001

A year or two ago, it seemed like everyone had heard about LDAP, and quite a few people were talking about it, but no one was really doing anything with it.

That seems to finally be changing, which is especially good for administrators and developers. LDAP can play a vital role in networks of all sizes, but like most new technology, it suffers from the Catch-22 of no one using it because it's not supported and developers not supporting it because no one is using it.

To understand why and how LDAP is going to be such an important tool in the life of a network administrator, it is necessary to understand what problems LDAP was developed to solve and how it will do so. This means it is also necessary to understand LDAP itself, both as a technology and as a tool.

In this introductory article, I hope to introduce LDAP and the concept of online directories, and explain why you might want them and what you can do with them. In later articles, I'll provide a more in-depth technical explanation of how to use LDAP, along with some example applications.

What is LDAP?

LDAP is the latest iteration in a rather lengthy development process beginning with the X.500 directory specification and its corresponding Directory Access Protocol (DAP) in the late 1980s and early 1990s. (For a more complete history of LDAP, and more information in general, see "Understanding and Deploying LDAP Directory Services", by T. Howes, M. Smith, and G. Good.)

DAP was a consistently difficult protocol to work with and implement, so easier protocols were developed with most of its functionality but significantly less complexity. Eventually, these versions were passed on to the IETF and OSI-DS and got merged into the Lightweight Directory Access Protocol, or LDAP, specification, first published as RFC 1487 in 1993. LDAP gained some widespread use in version 2, specified in RFC 1777.

LDAP is a protocol definition for accessing specialized databases

Because of the difficulty in truly separating the job of system administrator from the job of network administrator, and because there is often so much cross-over, especially in smaller environments, I will generally refer to people of either category as network administrators. I choose "network administrator" as the generic term because a quality system or network administrator is concerned with the entire network of devices or systems, not individual nodes. The term "network administrator" places more stress on viewing the network as a whole, making it a more appropriate term.

called directories. It is similar to SQL in that it is a language for interacting with databases without specifying a particular database. In fact, the back-end for LDAP directories is nearly always a more general RDBMS system, such as LDBM or Oracle.

Using LDAP to interact with a database does place constraints on that database, because of the assumptions the protocol makes and the specialized needs of a directory versus a standard relational database. But these constraints are necessary to be able to gain all of the desired features of a directory.

What is a directory, and why does my network need one?

Use of the word "directory" in this context may confuse people into thinking that most networks currently don't make use of directories, or that LDAP will be the only directory on the network. In actuality, directories are already a mainstay of life, especially in the computer world. Most people are familiar with talking about a phonebook or a map of the mall as a directory, but for some reason insist upon using the term "database" in computing, even when directory would be more specific and correct. As an example, I call Unix's system of storing user information the "passwd database," but that database easily qualifies as a directory.

At its most basic definition, a directory is any database specialized more for reading than for writing: The phonebook only comes out once a year, the mall directory is only changed when stores change, and the passwd database is only updated when user information is changed -- but all of that information is read frequently. The definition really does not get more specific than that because there are so many different information stores which can qualify as directories, although generally speaking a directory is much more likely to be searched than browsed. The listing most often referred to as a directory in the technical industry, a file system directory, also fits this definition, because the directory is read whenever a listed file is accessed in any way, but is only written when files are created or destroyed. Also, the directory is far more likely to be read when searching for a specific file rather than just browsing the listing.

Almost anyone involved in the development or maintenance of networked applications or services is already working with at least one directory: a system of maintaining user information. Nearly all services require some sort of authentication services, thus mandating that those same services maintain a user directory. Because of this, the most common form of online directory is for user information. Directories are useful for a larger variety of information than that, though. For instance, Unix systems maintain directories for services, groups, and many other data types, the Domain Name System (DNS) is a very specialized global directory for host-name-to-address correlation, and web directories used for navigating sets of web sites are springing up everywhere.

If I'm already using directories, why switch to LDAP?

We have already seen that almost any network uses a variety of directories, often for specific services. In database-speak, that means the directory data is not normalized, which means many pieces of data are stored in more than one place, and thus must be changed in more than one place when changes are necessary.

This is a problem for many reasons. The most obvious is that every time any information in any of these directories gets changed, all of the other directories must be hunted through to make that same change. This is not only difficult, it is often completely unmanageable -- witness the ease with which passwords for the same user in different services go out of sync.

Beyond that, every time two services implement their own versions of the same style of directory, there is significant redundant effort. Not only did the developers for each service have to develop their own directory, but now the managers of each service have to separately maintain the directories -- a single user of both services will almost certainly have a different user experience with each service, and it is nearly impossible to centrally manage these multiple directories.

Security is an even worse problem. Probably every developer and administrator is familiar with the headaches associated with user security. Are the passwords secure? Is the transport secure? Has the user really proved his/her identity? Did I accidentally leave a loophole to gain higher access? Will the user directory always be available? When multiple services implement separate directories for the same information, each of them must completely cover the security issues. Basic statistics virtually guarantee more security holes than a unified directory, and this also means that the services are likely to have differing and even conflicting security policies.

What you really want in a network is unification of directories, and this is exactly what LDAP was designed for. With this unification, you get data normalization, central management, consistent user experience, consistent management and security policies, fewer security holes, and less wasted development time.

Why should I trust LDAP to unify my directories?

LDAP was specifically designed to solve the problems caused by the proliferation of directories across a network, and there are seven aspects of its current implementation that give it that ability.

- **General-purpose design**
  Because LDAP was designed to be a general-purpose directory, it had to be extensible. It uses an object-oriented, inheritance-based schema definition, which provides for easy extension to any reasonable use. There is a base schema as part of the LDAP specification, and there are other de facto standards for various services. However, it is expected that most developers will extend the base schemas.

- **Protocol simplicity**
  One of the most important aspects of LDAP development, and that which caused it to be adopted in lieu of DAP, is that it is a simple protocol and is relatively simple to implement and work with. This is borne out by the fact that LDAP is supported by most major programming languages, including C, Java, and Perl, and either is supported or will be supported by most major operating systems, including Solaris, GNU/Linux, Microsoft Windows, and Mac OS.

- **Distributed architecture**
  Using data replication, it is possible to replicate all or part of an LDAP directory to physically separate locations, which allows for highly-available data and puts the data as close as necessary to the client. Using referrals, data mastery of portions of the directory can be distributed across different LDAP servers, thus allowing portions of an enterprise or project to have control over necessary data while maintaining a single authority over each piece of data.

- **Security**
  A large focus of LDAP development has been security, with version 3 of the LDAP protocol bringing significant improvements.

There are three basic aspects of securing the information in a directory: access, authentication, and authorization (AAA, or Triple-A). *Access* is the ability to connect to a service and can be restricted based on details like time of day or IP address, *authentication* is the ability to prove to the service that a client is a valid user, and *authorization* is the service providing or denying specific rights or capabilities to the client.

For secure access, LDAP supports Transport Layer Security (TLS), which can encrypt all communication between the client and server. For authentication, LDAP supports Simple Authentication and Security Layer (SASL), which allows the client and server to negotiate a (hopefully secure) authentication method.

TLS and SASL provide encryption capabilities but not control over access and authentication. LDAP actually provides the ability to control all three aspects of AAA through Access Control Lists (ACLs). ACLs can be used to grant access based on many different factors. They can be used to force specific types of authentication, and once the client is fully authenticated as a valid user, ACLs are used to authorize the user.

Unfortunately, the syntax of ACLs is not yet part of the LDAP specification. It seems likely that Netscape's implementation of ACLs will be accepted as the standard, but that has not happened yet and different LDAP servers may implement ACLs in different ways. However, this should not affect development or functioning of the client.

- **Open standard**
  Because LDAP is an open standard maintained by the IETF, it can be used by any developers, companies, or administrators without fear of being tied to proprietary protocols or specific vendors, and allows the choice of implementation to be based on project details rather than interoperability concerns. This also means that LDAP can progress according to the needs of the people who use it, rather than a corporation concentrating on profits or marketshare.

- **Server feature and schema requests**
  The LDAP specification states that LDAP clients can request the entire feature list and data schema from any LDAP server, thus allowing the client to vary its functionality according to that of the server, which should provide greater interoperability across different implementations and different versions of LDAP.

- **Internationalization**
  LDAP uses UTF-8 for internal string representations. This allows LDAP to store and manipulate any language of the world.

This is not an exhaustive list of all of the features of LDAP, but it details some of the most important aspects of the protocol. In fact, one of the reasons that LDAP is being adopted now is only marginally related to LDAP itself: The computer industry, especially in the area of network management, is ready for enterprise-wide directories. This is evidenced by them cropping up everywhere, from servers such as Netscape's Directory Server and Microsoft's Active Directory to clients such as email programs and operating systems all the way to standards specifications such as the Directory Enabled Networks (DEN) initiative.

I'm sold. How can I get some?

Unfortunately, LDAP isn't quite the panacea we all dream of today. The main problem is that it isn't as

supported as it could be. For instance, Sun's Solaris operating system provides support for LDAP as its naming service, but that support doesn't include any type of encryption, which is basically unacceptable. There are also still a surprising number of vendors and programs that don't provide support where they should, and many technology decision-makers either don't know what LDAP is or don't think it matters to them.

This points to another problem with the state of LDAP today: There is not much in the way of lay-person documentation or support. Yes, you can read the RFCs and source code, but high-level descriptions of what LDAP is and how it might be useful, now and in the future, are sadly lacking, especially in succinct formats (the best LDAP book available today is almost 800 pages long). Even for the applications that currently support LDAP, it is usually far more difficult to get them to work than it should be, which is causing a slow adoption of LDAP.

Then why would I even bother?

Although it is easy to take a gloomy view on the future of LDAP -- and certainly its present -- the fact is that more vendors, applications, languages, and operating systems are supporting LDAP, and more organizations are depending on it for management of key information. LDAP directories are great for organizations that do a lot of web development or custom application development. Although the industry support isn't the best right now, most vendors either currently partially or fully support LDAP or are talking about doing so. And frankly, LDAP really is a simple protocol and has APIs for any language. If you are developing web applications, or just need a good directory to store information in, chances are you should at least look at LDAP -- you may like what you find.

So now is the time to begin learning about LDAP and trying to understand what role it can play in what you do, whether you function as a developer, administrator, decision-maker, or all three. The longer it takes our industry to fully support LDAP as a standard for directories, the more time we all have to spend maintaining redundant and often inadequate information stores and methods of interacting with them.

Where we go from here

Next time we'll cover the LDAP protocol in more depth, and begin to develop our LDAP expertise by building a basic LDAP directory.

Luke A. Kanies is an independent consultant and researcher specializing in Unix automation and configuration management.

Return to ONLamp.com.

Introduction to LDAP

Brad Marshall
brad.marshall@member.sage-au.org.au
http://quark.humbug.org.au

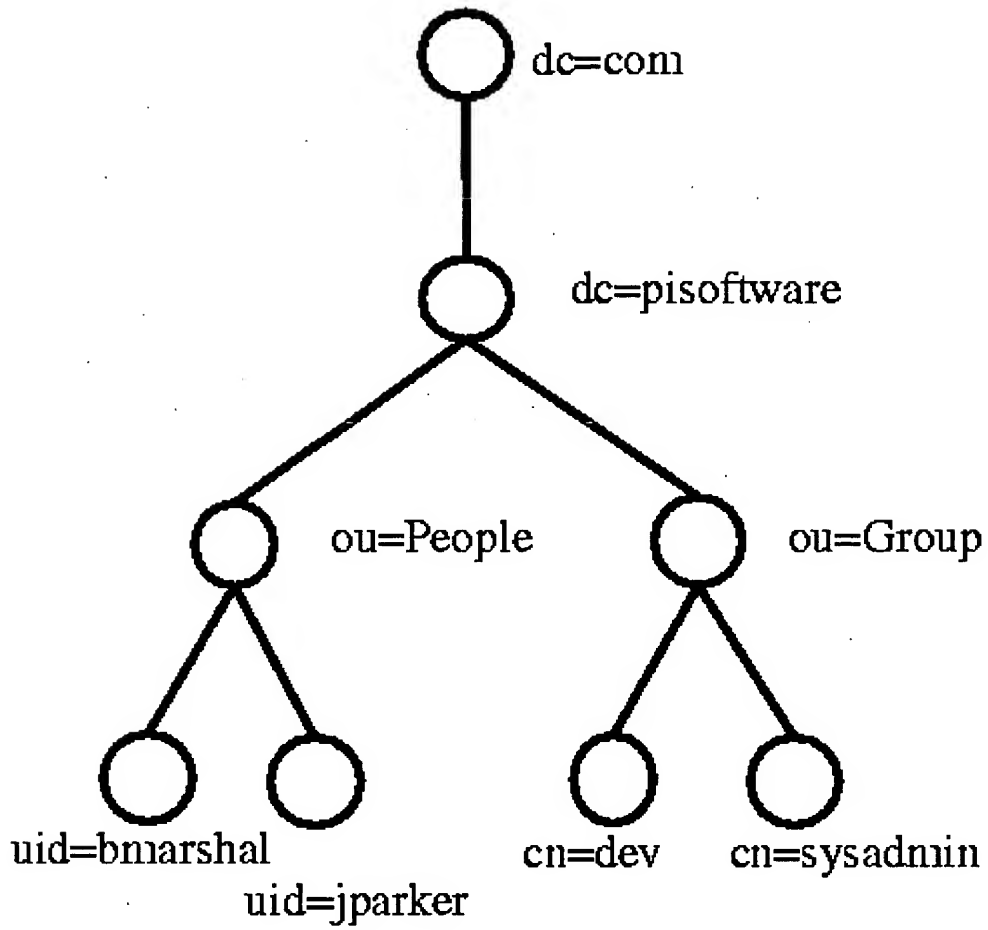
Other LDAP related tutes and papers are available.

Index

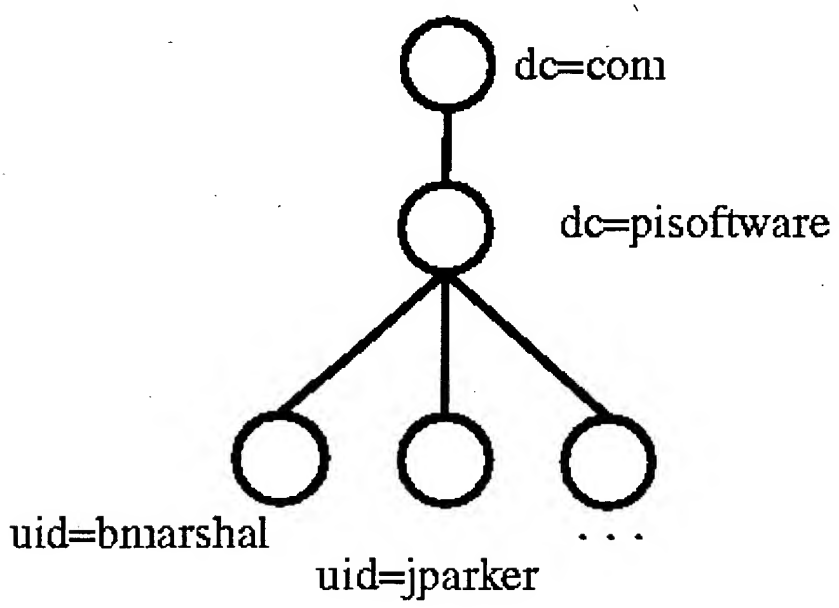
What is LDAP

- Lightweight Directory Access Protocol
- Based on X.500
- Directory service (RFC1777)
- Stores attribute based data
- Data generallly read more than written to
    - No transactions
    - No rollback
- Hierarchical data structure
    - Entries are in a tree-like structure called Directory Information Tree (DIT)

Hierachial

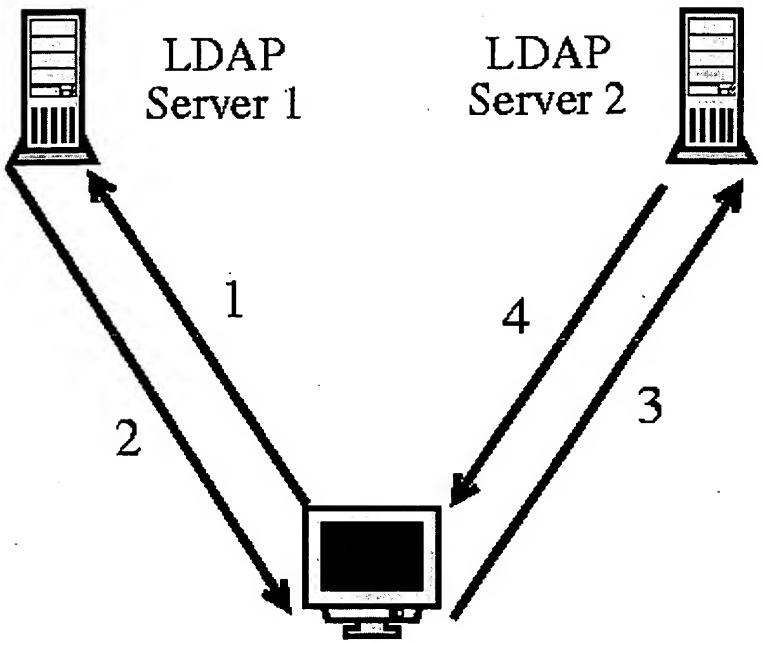


Flat



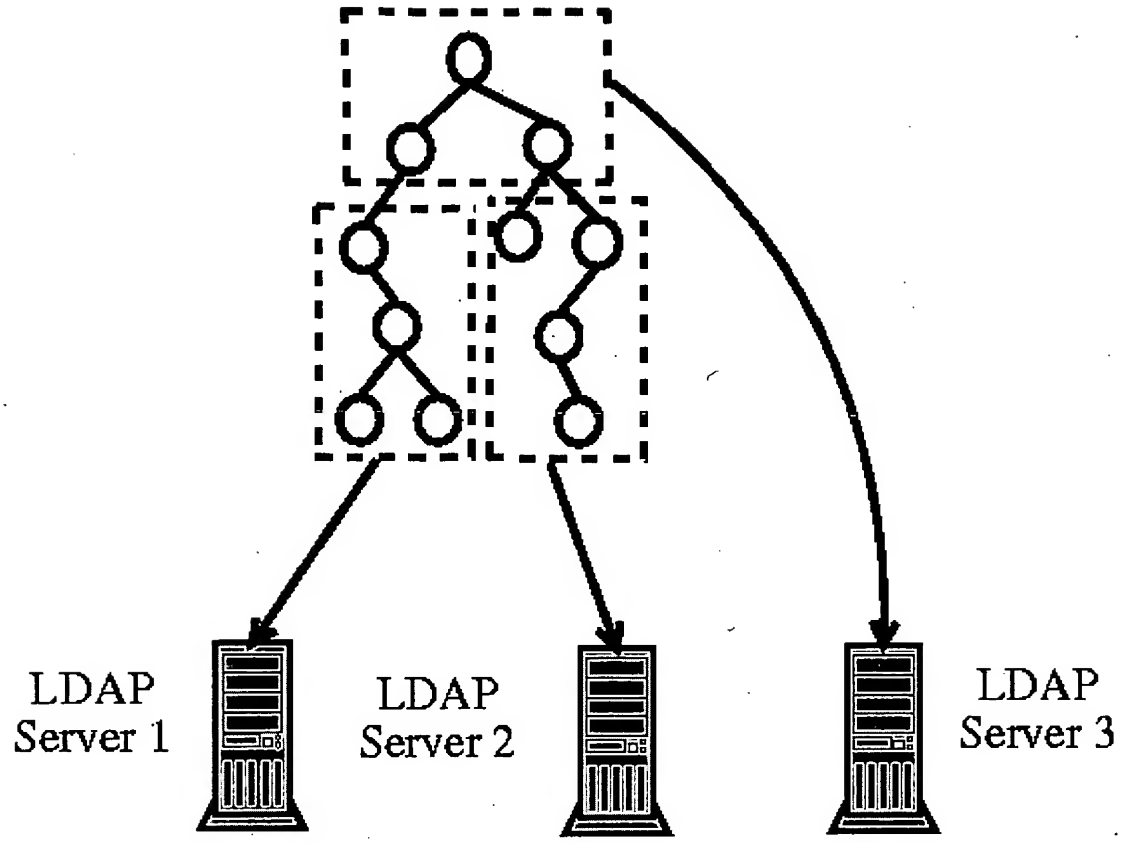
- Client-server model
- Consistant view of data
  - Answers request
  - Refer to server with answer

Referrals



1. Client requests information
2. Server 1 returns referral to server 2
3. Client resends request to server 2
4. Server 2 returns information to client

Global View



- Based on entries
  - Collection of attributes
  - Has a distinguished name (DN) - like domain name

Acroynms

LDAP
: Lightweight Directory Access Protocol

DN
: Distinguish Name

RDN
: Relative Distinuished Name

DIT
: Directory Information Tree

LDIF
: LDAP Data Interchange Format

OID
: Object Identifier

LDIF

- LDAP Data Interchange Format
  - Represents LDAP entries in text
  - Human readable format

    - Allows easy modification of data
- ldbmcat converts ldbm database to ldif
- ldif2ldbm converts ldif back to ldbm database
- Example extract

```
dn: uid=bmarshal,ou=People,dc=pisoftware,dc=com
uid: bmarshal
cn: Brad Marshall
objectclass: account
objectclass: posixAccount
objectclass: top
loginshell: /bin/bash
uidnumber: 500
gidnumber: 120
homedirectory: /mnt/home/bmarshal
gecos: Brad Marshall,,,,
userpassword: {crypt}KDnOoUYN7Neac
```

Schema

- Set of rules that describes what kind of data is stored
- Helps maintain consistancy and quality of data
- Reduces duplication of data
- Object class attribute determines schema rules the entry must follow
- Schema contains the following:
    - Required attributes
    - Allowed attributes
    - How to compare attributes
    - Limit what the attributes can store - ie, restrict to integer etc
    - Restrict what information is stored - ie, stops duplication etc

Attribute abbreviations

See RFC2256

uid
: User id

cn
: Common Name

sn
: Surname

l
: Location

ou
: Organisational Unit

o
: Organisation

dc
: Domain Component

st
: State

c
: Country

Search Filters

- Criteria for attributes that must be fulfilled for entry to be returned
- Base dn = base object entry search is relative to
- Prefix notation
- Standards
  - RFC 1960: LDAP String Representation of Search Filters
  - RFC 2254: LDAPv3 Search Filters
- Operators
  - & = and
  - | = or
  - ! = not
  - ~= = approx equal
  - >= = greater than or equal
  - <= = less than or equal
  - * = any
- Eg
  - (objectclass=posixAccount)
  - (cn=Mickey M*)
  - (|(uid=fred)(uid=bill))
  - (&(|(uid=jack)(uid=jill))(objectclass=posixAccount))

LDAP URL

Definition taken from RFC1959

```
<ldapurl> ::= "ldap://" [ <hostport> ] "/" <dn> [ "?" <attributes>
                        [ "?" <scope> "?" <filter> ] ]
<hostport> ::= <hostname> [ ":" <portnumber> ]
<dn> ::= a string as defined in RFC 1485
<attributes> ::= NULL | <attributelist>
<attributelist> ::= <attributetype>
                  | <attributetype> [ "," <attributelist> ]
<attributetype> ::= a string as defined in RFC 1777
<scope> ::= "base" | "one" | "sub"
<filter> ::= a string as defined in RFC 1558
```

Explanations:

DN
: Distinguished name

Attribute list
: List of attributes you want returned

Scope
: base = base object search
: one = one level search
: sub = subtree search

Filter
: Standard LDAP search filter

Examples:

- ldap://foo.bar.com/dc=bar,dc=com
- ldap://argle.bargle.com/dc=bar,dc=com??sub?uid=barney
- ldap://ldap.bedrock.com/dc=bar,dc=com?cn?sub?uid=barney

LDAP command line tools

ldapadd, ldapmodify
: Used to add or modify ldap entries

```
$ ldapmodify -r -D 'cn=foo,dc=bar,dc=com' -W < /tmp/user.ldif
```

ldapdelete
: Used to delete entries

```
$ ldapdelete -D 'cn=foo,dc=bar,dc=com' -W 'cn=user,dc=bar,dc=com'
```

ldapsearch
: Used to search ldap servers

```
$ ldapsearch -L -D 'cn=foo,dc=bar,dc=com' 'objectclass=posixAccount'
```

Installing and Configuring LDAP

Servers

- Slapd
  - University of Michigan
  - Openldap
- Netscape Directory Server
- Microsoft Active Directory (AD)
- Novell Directory Services (NDS)
- Sun Directory Services (SDS)
- Lucent's Internet Directory Server (IDS)

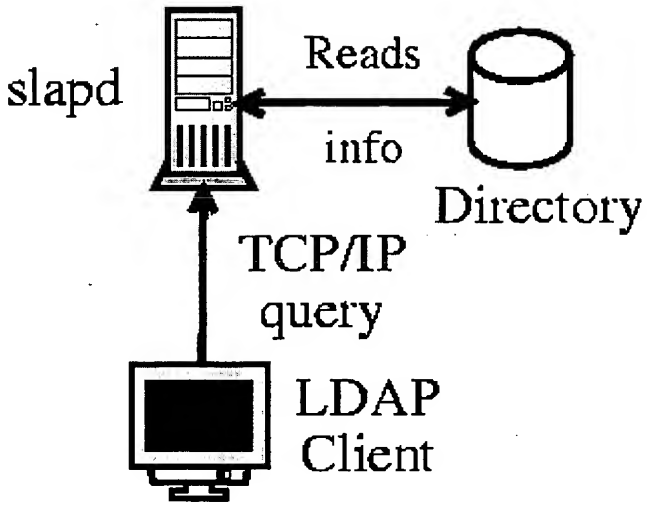
Openldap

LDAP Server architecture

- LDAP daemon called slapd
  - Choice of databases
    - LDBM - high performance disk based db
    - SHELL - db interface to unix commands
    - PASSWORD - simple password file db
    - SQL - mapping sql to ldap (in OpenLDAP 2.x)
  - Multiple database instances
  - Access control
  - Threaded

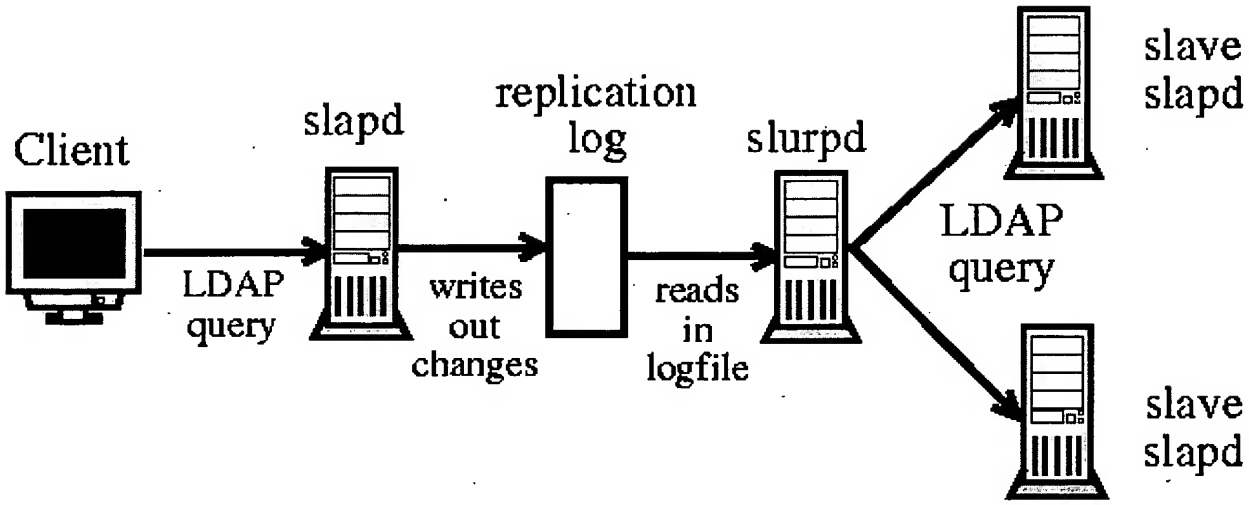
- Replication

LDAP Architecture



- Replication daemon called slurpd
  - Frees slapd from worrying about hosts being down etc
  - Communicates with slapd through text file

Replication Architecture



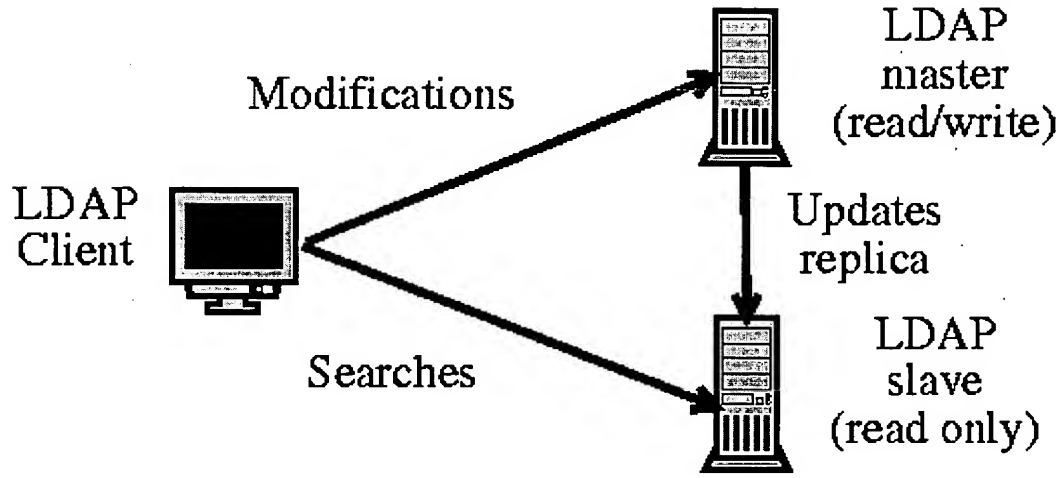
Replication

- Increases:
  - Reliability - if one copy of the directory is down
  - Availability - more likely to find an available server
  - Performance - can use a server closer to you

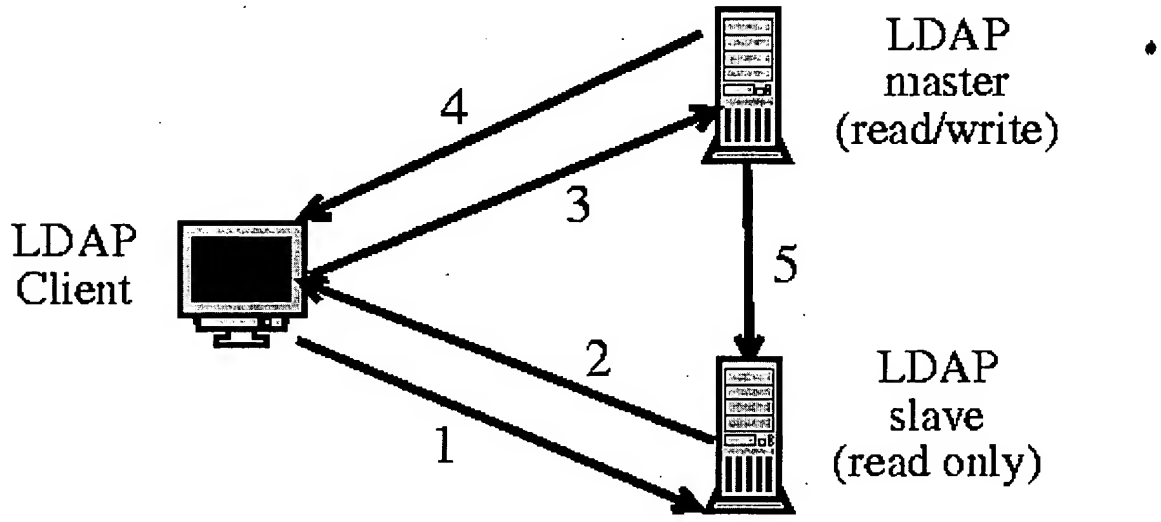
- Speed - can take more queries as replicas are added
- Temporary inconsistances are ok
- Having replicas close to clients is important - network going down is same as server going down
- Removes single point of failure

Replication Options

a. All modifications go to the master LDAP server



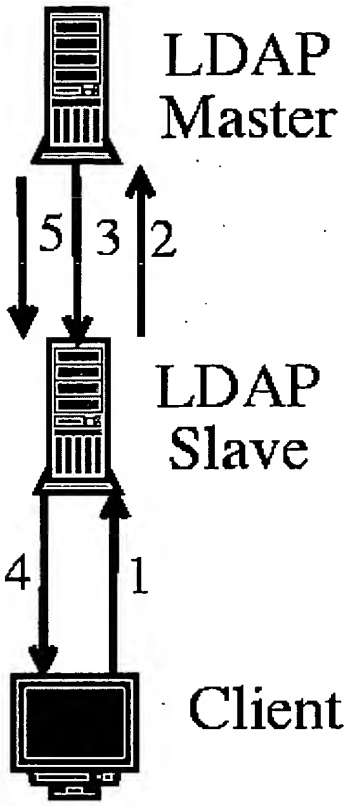
b. Using referrals



1. Client sends modification to replica
2. Replica returns referral to master
3. Client resubmits modification to master
4. Master returns results to client

5. Master updates replica with change

c. Using chaining



1. Client sends modification to replica
2. Replica forwards request to master
3. Master returns result to replica
4. Replica forwards result to client
5. Master updates replica

Example slapd.conf

```
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
include         /etc/openldap/slapd.at.conf
include         /etc/openldap/slapd.oc.conf
schemacheck     off

pidfile         /var/run/slapd.pid
argsfile        /var/run/slapd.args

defaultaccess read

access to attr=userpassword
   by self write
   by * read

access to *
   by self write
   by dn=".+" read
   by * read
```

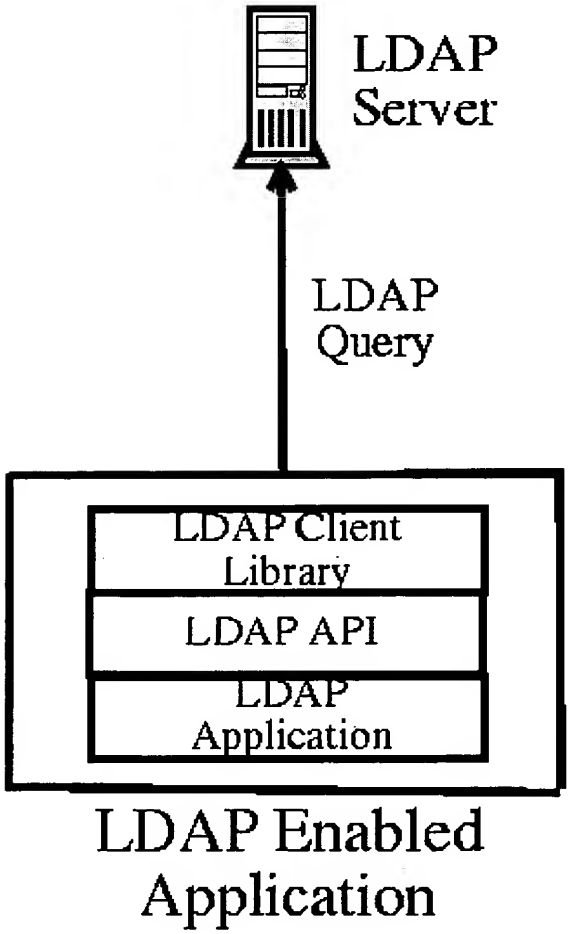
```
#########################################################################
# ldbm database definitions
#########################################################################

database        ldbm
suffix          "dc=pisoftware, dc=com"
rootdn          "cn=Manager, dc=pisoftware, dc=com"
rootpw          {crypt}lAn4J@KmNp9
replica host=cox.staff.plugged.com.au:389
    binddn="cn=Manager,dc=pisoftware,dc=com"
    bindmethod=simple credentials=secret
    replogfile /var/lib/openldap/replication.log
# cleartext passwords, especially for the rootdn, should
# be avoid.  See slapd.conf(5) for details.
directory       /var/lib/openldap/
```

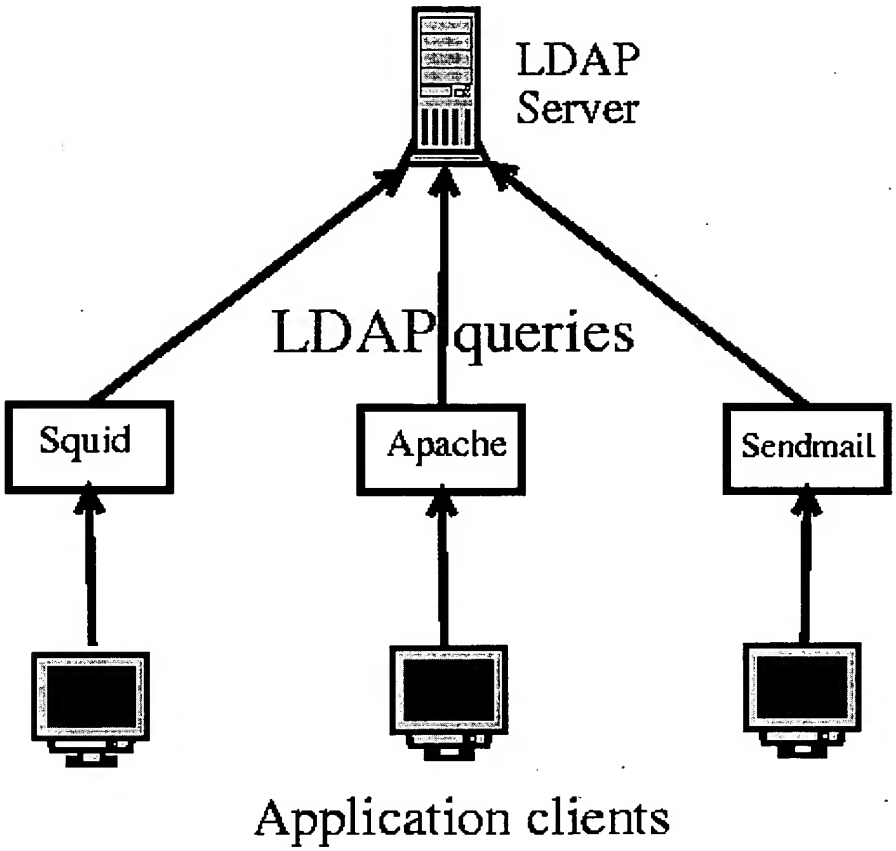
slapd.conf ACLs

LDAP Applications

Application Architecture



Using Multiple Applications



System Authentication

Uses RFC2307

Migration

Used PADLs MigrationTools

Script	Migrates
migrate_fstab.pl	/etc/fstab
migrate_group.pl	/etc/group
migrate_hosts.pl	/etc/hosts
migrate_networks.pl	/etc/networks
migrate_passwd.pl	/etc/passwd
migrate_protocols.pl	/etc/protocols
migrate_rpc.pl	/etc/rpc
migrate_services.pl	/etc/services

These scripts are called on the appropriate file in /etc in the following manner:

```
# ./migrate_passwd.pl /etc/passwd ./passwd.ldif
```

The migration tools also provide scripts to automatically migrate all configuration to LDAP, using migrate_all_{online,offline}.sh. See the README distributed with the package for more details.

Example LDIF

```
dn: uid=bmarshal,ou=People,dc=pisoftware,dc=com
uid: bmarshal
cn: Brad Marshall
objectclass: account
objectclass: posixAccount
objectclass: top
loginshell: /bin/bash
uidnumber: 500
gidnumber: 120
homedirectory: /mnt/home/bmarshal
gecos: Brad Marshall,,,,
userpassword: {crypt}aknbKIfeaxs


dn: cn=sysadmin,ou=Group,dc=pisoftware,dc=com
objectclass: posixGroup
objectclass: top
cn: sysadmin
gidnumber: 160
memberuid: bmarshal
memberuid: dwood
memberuid: jparker
```

Installation

Install from PADL

- pam_ldap
- nss_ldap

/etc/ldap.conf

```
BASE        dc=foo,dc=com
HOST        ldap.server.com
pam_crypt   local
```

/etc/nsswitch.conf

Add ldap to the passwd, shadow and group entries in /etc/nsswitch.conf. Be aware of the effects of putting it first or last.

/etc/pam.d

Need similar for every app you want to use ldap

/etc/pam.d/ssh

From RedHat 6.2

```
#%PAM-1.0
auth       sufficient   /lib/security/pam_ldap.so
auth       required     /lib/security/pam_pwdb.so shadow nullok try_first_pass
auth       required     /lib/security/pam_nologin.so
account    sufficient   /lib/security/pam_ldap.so
account    required     /lib/security/pam_pwdb.so
password   required     /lib/security/pam_cracklib.so
password   sufficient   /lib/security/pam_ldap.so
password   required     /lib/security/pam_pwdb.so shadow nullok use_authtok
session    sufficient   /lib/security/pam_ldap.so
session    required     /lib/security/pam_pwdb.so
```

Apache user auth

- Download mod_auth_ldap.tar.gz from http://www.muquit.com/muquit/software/mod_auth_ldap/mod_auth_ldap.html
- Install either as a DSO or by compiling in - see webpage for more details
- Add the following to httpd.conf

```
<Directory "/var/www/foo">
Options Indexes FollowSymLinks
AllowOverride None
order allow,deny
allow from all
AuthName "RCS Staff only"
AuthType Basic
LDAP_Server ldap.server.com
LDAP_Port 389
Base_DN "dc=server,dc=com"
UID_Attr uid
#require valid-user
require user foo bar doe
#require roomnumber "C119 Center Building"
#require group cn=sysadmin,ou=Group,dc=server,dc=com
</Directory>
```

Squid ACLs

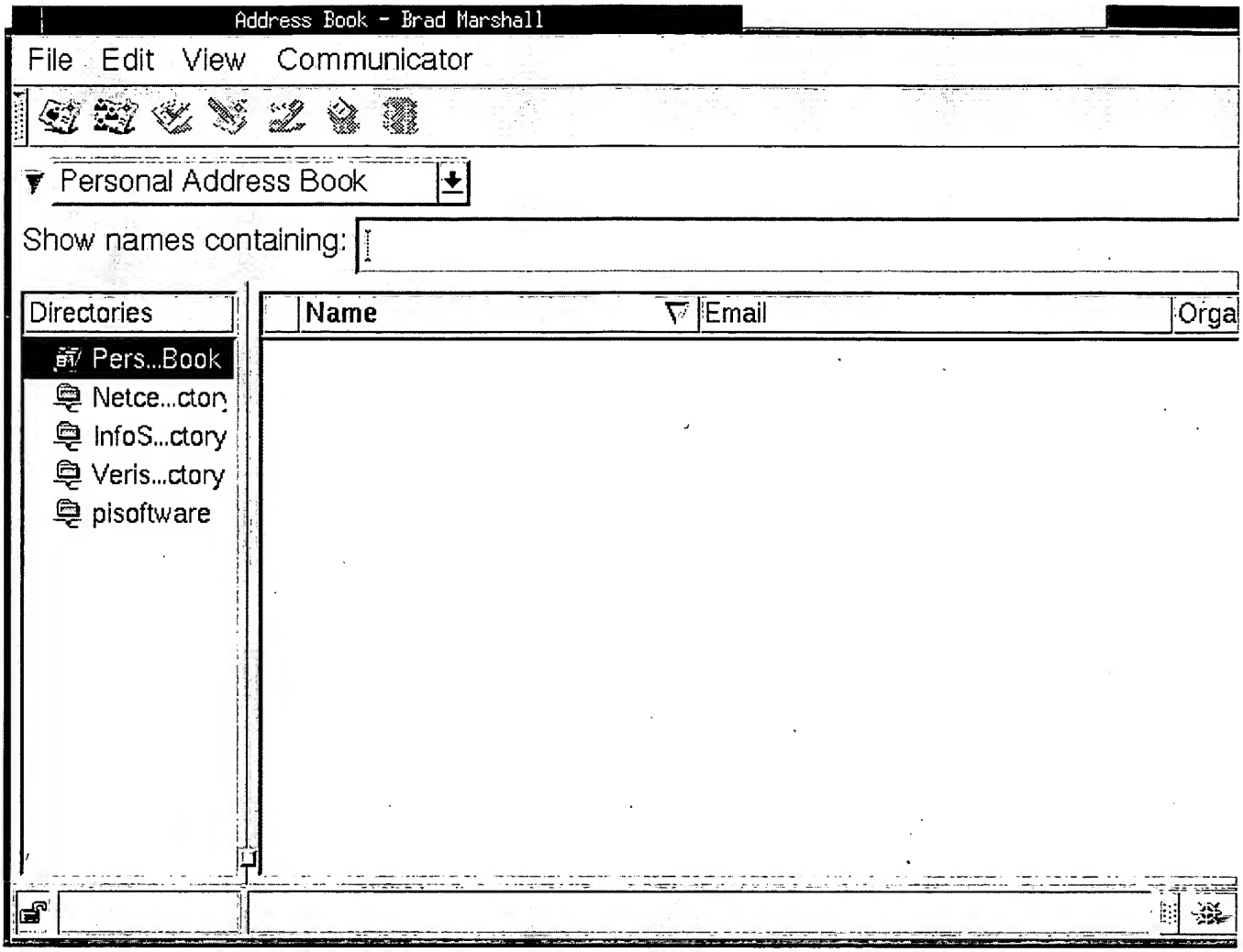
- Compile ldap_auth.c from http://www.uia.ua.ac.be/u/dbruyne/squid-ldap/
- Add the following to squid.conf:

```
authenticate_program /usr/local/squid/bin/ldap_auth
authenticate_options ldap.yourdomain.com 389 dc=yourdomain,dc=com uid
authenticate_children 2
```

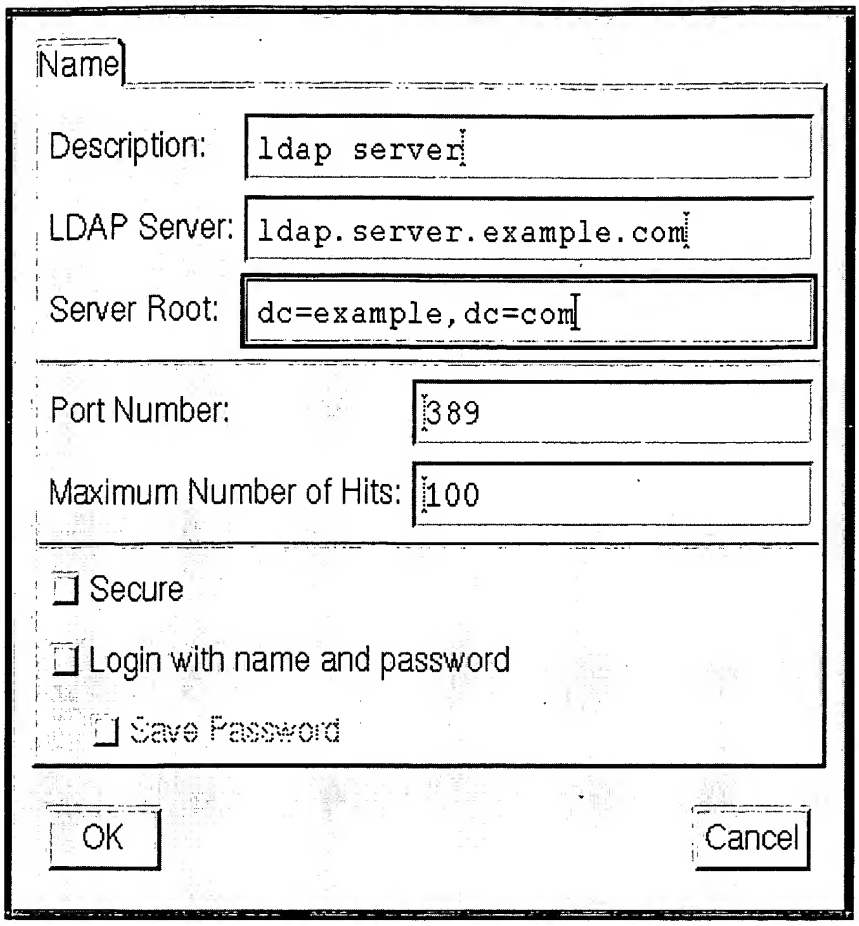
- Restart squid

Netscape Address book

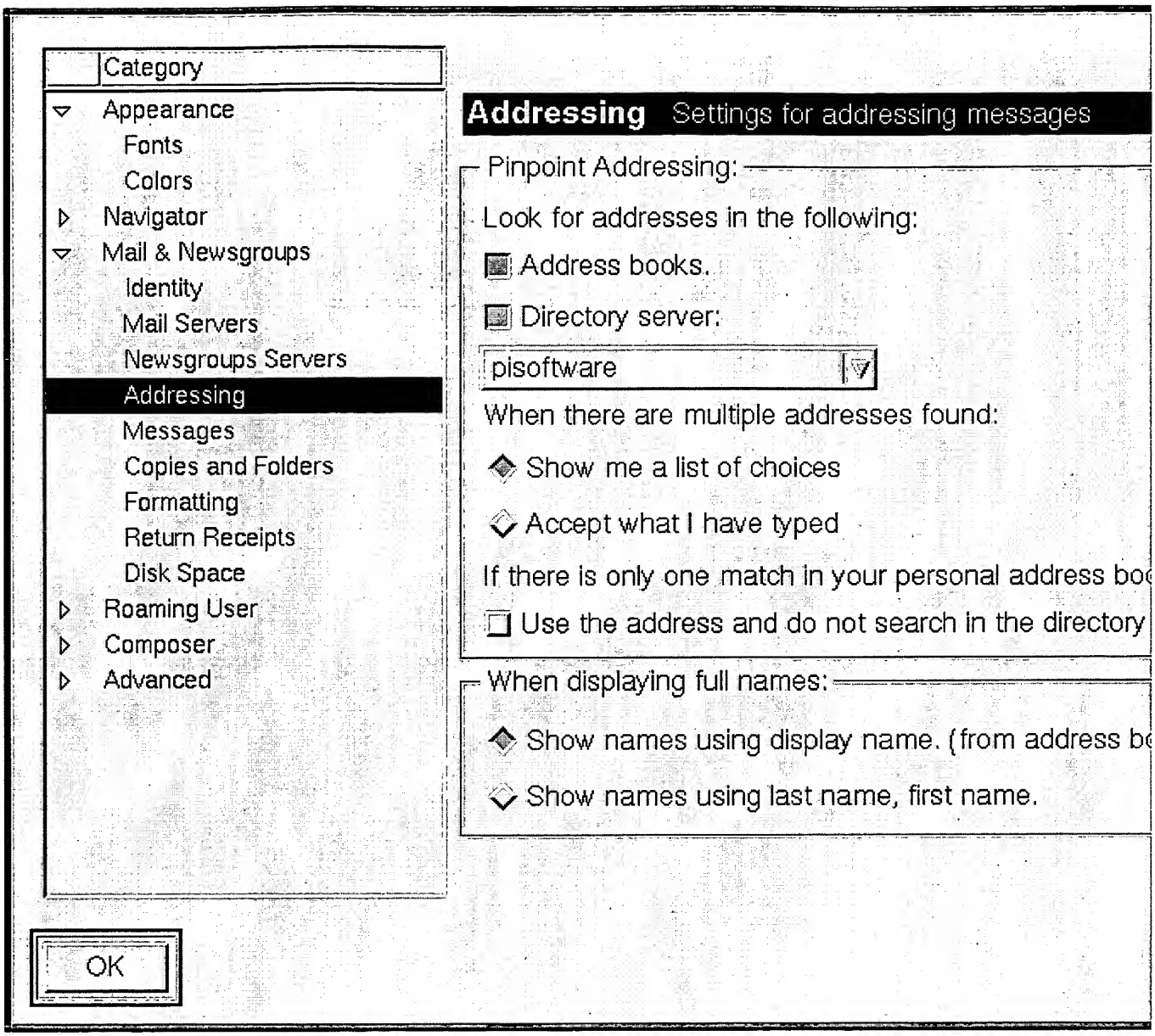
To add a LDAP server to the Netscape Address book:



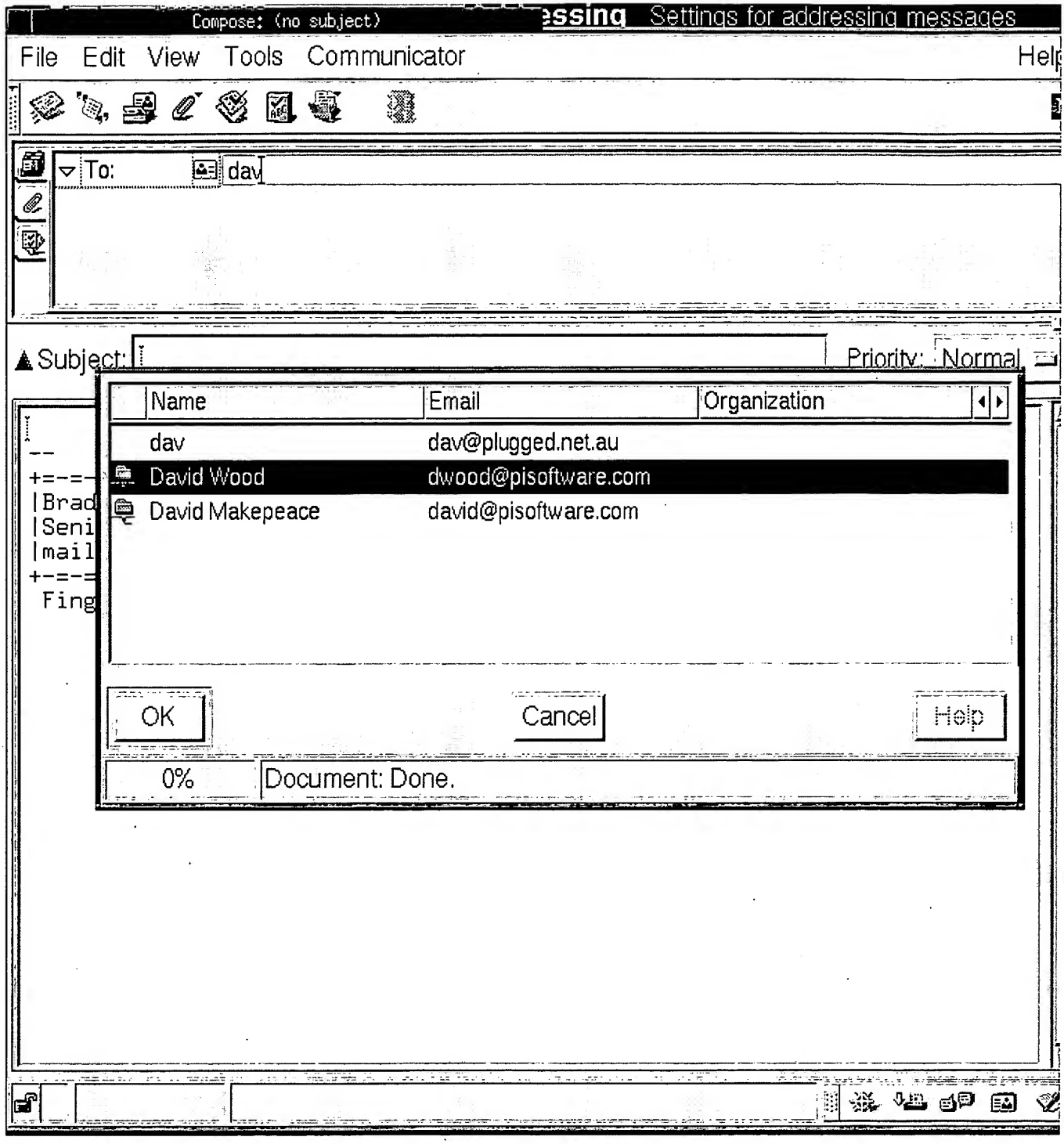
Netscape Address Book



Add a new directory to the address book



Add the new directory into the addressbook search



Example of searching directory server

The email address returned is the contents of the `mail' attribute.

Mutt Address book

```
#!/usr/bin/perl -w
# by Ben Collins , butchered by Marco d'Itri
```

```
# Hacked by Brad Marshall  for use at PI
# to use, add to ~/.muttrc:
#  set query_command="/mnt/home/linux/bin/pi-ldap-query %s"

use strict;

my @attrs = qw(sn cn uid);
my $base = 'ou=People, dc=pisoftware, dc=com';
my $server = 'morris';
my $port = 389;

die "Usage: $0 [...]\n" if not $ARGV[0];

eval 'require Net::LDAP;';
die "Could not load Net::LDAP: $@\n" if $@;

my $ldap = Net::LDAP->new($server, port => $port) or
        die "Could not contact LDAP server $server:$port";
$ldap->bind or die 'Could not bind';

my @results = ();

foreach my $search (@ARGV) {
        my $query = join '', map { "($_=*$search*)" } @attrs;
    my $mesg = $ldap->search(base => $base, filter => "(|$query)")
                or die 'Failed search';
    foreach my $entry ($mesg->entries) {
                my $uid = $entry->get('uid');
                next unless (defined $uid);
                my $fname = $entry->get('cn');
                #my $lname = $entry->get('sn');
                my $mail = $entry->get('mail');
                push @results,
                        "<$$mail[0]>\t$$fname[0]\tPI\n";
    }
}

$ldap->unbind;

print 'LDAP query: found ', scalar @results, "\n", @results;
exit 1 unless @results;
```

Sendmail routing

http://sendmail.net/?feed=ldaprouting http://sendmail.net/?feed=rfc1777

Samba

http://www.unav.es/cti/ldap-smb-howto.html

Netscape Roaming access

http://www.linuxworld.com/linuxworld/lw-1999-09/lw-09-ldap-netscape.html

Programming LDAP

Perl

Basic Query

```
#!/usr/bin/perl -w

use strict;
use Net::LDAP;

my($ldap) = Net::LDAP->new('ldap.staff.plugged.com.au') or die "Can't bind to ldap:

$ldap->bind;

my($mesg) = $ldap->search( base => "dc=pisoftware,dc=com",
                                        filter => '(objectclass=*)');

$mesg->code && die $mesg->error;

my($entry);

map { $_->dump } $mesg->all_entries;
# OR
foreach $entry ($mesg->all_entries) { $entry->dump; }

$ldap->unbind;
```

Adding Entries

```
#!/usr/bin/perl -w

use strict;
use Net::LDAP;

my $root = "dc=pisoftware,dc=com";
my $manager = "cn=Manager,$root";
my $password = 'secret';

my $groupdn = "cn=test,ou=Group,$root";
my $uid = "test";

my($ldap) = Net::LDAP->new('ldap.staff.plugged.com.au') or die "Can't bind to ldap:

$ldap->bind(
            dn       => $manager,
            password => $password,
        );

#$ldap->modify( $groupdn, add => { memberuid => $uid } );
$result = $ldap->add( dn => $groupdn,
                      attr => [ 'cn' => 'Test User',
                                'sn' => 'User',
                                'uid' => 'test',
```

```
                                        ];

$result->code && warn "failed to add entry: ", $result->error;

$ldap->unbind;
```

Deleting Entries

```
#!/usr/bin/perl -w

use strict;
use Net::LDAP;

my $root = "dc=pisoftware,dc=com";
my $manager = "cn=Manager,$root";
my $password = 'secret';

my $groupdn = "cn=test,ou=Group,$root";
my $uid = "test";

my($ldap) = Net::LDAP->new('ldap.staff.plugged.com.au') or die "Can't bind to ldap:

$ldap->bind(
            dn       => $manager,
            password => $password,
        );

$ldap->delete( $groupdn );
$ldap->unbind;
```

Modifying Entries

```
$ldap->modify( $dn,
        changes => [
                add     => [ sn => 'User' ],                    # Add sn=User
                delete  => [ faxNumber => []],                  # Delete all fax num
                delete  => [ telephoneNumber => ['911']],       # delete phone numbe
                replace => [ email => 'test@pisoftware.com']    # change email addre
        ]
);
```

Real World Examples

Rsync Backup Scripts

Given an ldap server with entries for each machine, does a rsync backup from them using the module names given in the rsync attribute. This allows automatic backing up of hosts on whatever schedule the script is run from cron.

Schema

```
objectclass machine
```

```
        requires
                                objectClass,
                                hostname,
                                cpu,
                                ram,
                                usage
        allows
                                rsync
```

LDIF Extract

```
dn: cn=carmack,ou=Machines,dc=pisoftware,dc=com
hostname: carmack
objectclass: top
objectclass: machine
usage: workstation
rsync: etc
cpu: PIII 550
ram: 256M
```

Code

```
#!/usr/bin/perl -w
# ------------------------------------------------------------
# script:  ldaprsync.pl
# Author:  Brad Marshall (bmarshal@pisoftware.com)
# Date:    20000801
#
# Purpose: Rsync certain modules from hosts in ldap
#
# Copyright (c) 2000 Plugged In Software Pty Ltd.  All rights reserved.

use strict;
use Net::LDAP;

my %config = ( "destdir" => "/opt/hosts",
                                "rsyncoptions" => "--compress --archive --one-file-s

my($ldap) = Net::LDAP->new('ldap.staff.plugged.com.au') or die "Can't bind to ldap:

$ldap->bind;

my($mesg) = $ldap->search( base => "dc=pisoftware,dc=com",
                                                filter => '(objectclass=machine)');

$mesg->code && die $mesg->error;

my($entry,$attr);

my(%results);
foreach $entry ($mesg->entries) {
        #print "DN = ",$entry->dn,"\n";
        #my($tmpcn) = $entry->get('uid');
        my(@rsync) = $entry->get('rsync');
        my(@hostname) = $entry->get('hostname');
        my($host) = join(" ", @hostname);
```

```
        foreach my $src (@rsync) {
                print "Getting $host $src\n";

                my $direct;
                if ($src =~ /^(.*)\//g) {
                        $direct = $1;
                } else {
                        $direct = $src;
                }
                my $dir = "$config{destdir}/$host/$direct";
                if ( ! -d $dir ) {
                        system("mkdir -p $dir");
                }
                my @ary = split /\s+/, "rsync $config{rsyncoptions} $host\::$src $di
                system(@ary) == 0 or warn "system @ary failed: $?\n";
                #print "@ary\n" or warn "system @ary failed: $?\n";
        }
#       foreach $attr ($entry->attributes) {
#               print $attr,": ", join(" ", $entry->get($attr)), "\n";
#       }
#       print "\n";
}

$ldap->unbind;
```

Squid Authentication

Dumps the username and password pairs from LDAP, optionally using a httppassword attribute instead of the userpassword attribute, if it exists. Useful for doing proxy authentication if you don't want to, or can't, use existing ldap authentication methods.

```
#!/usr/bin/perl -w
# ------------------------------------------------------------
# script:  ldapdumpsquid.pl (based on ldapdumpusers.pl)
# Author:  Brad Marshall (bmarshal@pisoftware.com)
# Date:    20000717
#
# Purpose: Dumps the users into a htaccess type file
#          Useful for apache and squid.
#
# Copyright (c) 2000 Plugged In Software Pty Ltd.  All rights reserved.

use strict;
use Net::LDAP;

my($ldap) = Net::LDAP->new('ldap.staff.plugged.com.au') or die "Can't bind to ldap:

$ldap->bind;

my($mesg) = $ldap->search( base => "dc=pisoftware,dc=com",
                                        filter => '(objectclass=account)');

$mesg->code && die $mesg->error;

my($entry,$attr);

my(%results);
```

```
foreach $entry ($mesg->entries) {
# postgres:!!:113:113:PostgreSQL Server:/var/lib/pgsql:/bin/sh
# uid:userpassword:uidnumber:gidnumber:gecos:homedirectory:loginshell

        my($tmpcn) = $entry->get('uid');
        my($tmpnum) = $entry->get('uidnumber');
        my($tmpgid) = $entry->get('gidnumber');
        my($tmppassword) = $entry->get('httppassword');
        #defined $tmppassword and do { print ref $tmppassword,"\n"; print $tmppasswo
        defined $tmppassword or ($tmppassword) = $entry->get('userpassword');
        if ($tmppassword) {
                $tmppassword =~ s/^{crypt}//;
        }
        #my($tmpgecos) = $entry->get('gecos') || [ "Plugged In User" ];
        my($tmpgecos) = $entry->get('cn');
        $tmpgecos ||= "";
        my($tmphome) = $entry->get('homedirectory');
        my($tmpshell) = $entry->get('loginshell');
        $tmpshell ||= "";

        if (! $tmppassword) {
                $tmppassword = "x";
        }
        if ($tmppassword eq "x") {
                next;
        }
        if ($tmppassword =~ /^\*/) {
                next;
        }

        if ($tmpcn =~ /(wallppp)|(order)|(helpdesk)|(javadoc)|(postgres)|(nocol)|(ba
                next;
        }
        my($tmpstr) = "$tmppassword";
        #print "$tmpcn\n";
        #if ($tmpnum => 500) {
        #       $results{$tmpcn} = $tmpstr;
        #}
        $results{$tmpcn} = $tmpstr;
}

my($foo);
foreach $foo (sort keys %results) {
        #printf "%3d\t%s\n",$results{$foo}, $foo;
        print "$foo:$results{$foo}\n";
}

$ldap->unbind;
```

User Admin Scripts

- ldapadduser.pl
- ldapdeluser.pl
- ldapdumpusers.pl
- ldaplistgroups.pl

ldapadduser.pl

```
#!/usr/bin/perl -w
# ------------------------------------------------------------
# script:  ldapadduser.pl
# Author:  Brad Marshall (bmarshal@pisoftware.com)
# Date:    20000203
#
# Purpose: Adds a user to LDAP
#
# Copyright (c) 2000 Plugged In Software Pty Ltd.  All rights reserved.

# TODO
# x Check username ( >= 8 chars, all alpha)
# x Check uid number (not negative, less than 65536, etc)
# x Handle ^C's etc gracefully
# x Check automatically generated userid / groupid
# x Check username / uid not already used
#   Get manager's password & bind as manager
#   Display information, ask if ok before adding

# Modules
use strict;
use Net::LDAP;
use Getopt::Std;
use Term::ReadKey;

use vars qw($opt_c $opt_d $opt_g $opt_G $opt_s $opt_u);

# Variables
my($username);
my($homedirectory);
my($gid);
my($gidnumber);
my($gidname);
my($groups);
my($loginshell);
my($uidnumber);
my($uid);
my($cn);
my($gecos);
my($dn);
my($result);
my($gidref);

my($root) = "dc=pisoftware,dc=com";
my($host) = "ldap.staff.plugged.com.au";
my(@objectclass) = [ 'account', 'posixAccount', 'top' ];
my($manager) = "cn=Manager,$root";
my $exception = 0;

# Signal handlers
$SIG{'INT'} = $SIG{'QUIT'} = $SIG{'HUP'} = sub { $exception=1; };

# useradd [-c comment] [-d home_dir]
#               [-e expire_date] [-f inactive_time]
#               [-g initial_group] [-G group[,...]]
#               [-m [-k skeleton_dir] | -M] [-p passwd]
#               [-s shell] [-u uid [ -o]] [-n] [-r] login

# Handle the command line options
```

```
getopts('c:d:g:G:s:u:');

if (! $ARGV[0]) {
        print "$0: $0 [-c comment] [-d directory] [-g default group] [-G groups, ..]
        exit 1;
}

if ($ARGV[0]) {
        $uid = $ARGV[0];
} else {
        die "Sorry, you must specify a login";
}
if ($opt_c) {
        $cn = $opt_c;
} else {
        $cn = "Plugged In Linux User";
}
$gecos = $cn;
if ($opt_d) {
        $homedirectory = $opt_d;
} else {
        $homedirectory = "/mnt/home/" . $uid;
}
if ($opt_g) {
        $gid = $opt_g;
        $gidref = &findgid($gid);

        $gidnumber = $gidref->[0];
        $gidname = $gidref->[1];

        print "number = $gidnumber, name = $gidname\n";
} else {
        die "Sorry, you must specify a default group for this user";
}
if ($opt_G) {
        $groups = $opt_G;
        print "\$groups = $groups\n";
}
if ($opt_s) {
        $loginshell = $opt_s;
} else {
        $loginshell = "/bin/bash";
}
if ($opt_u) {
        $uidnumber = $opt_u;
} else {
        # find next available uid
        $uidnumber = &finduid;

        print "uid = $uidnumber\n";
}

# Check the uid is valid
&checkuid;

print "Please enter LDAP Managers password: ";

ReadMode 'noecho';
my $password = ReadLine 0;
chomp $password;
```

```
ReadMode 'normal';
print "\n";

print "cn $cn\n";
print "uid $uid\n";
print "objectClass @objectclass\n";
print "uidNumber $uidnumber\n";
print "gidNumber $gidnumber\n";
print "loginShell $loginshell\n";
print "homeDirectory $homedirectory\n";
print "gecos $gecos\n";

# Add the user to all the groups in $groups
&addgroups;
#exit 0;

#dn: uid=bmarshal,ou=People,dc=pisoftware,dc=com
#uid: bmarshal
#cn: Brad Marshall
#objectClass: account
#objectClass: posixAccount
#objectClass: top
#userPassword: {crypt}j26bYFB8xQYLE
#loginShell: /bin/bash
#uidNumber: 500
#gidNumber: 120
#homeDirectory: /mnt/home/bmarshal
#gecos: Brad Marshall,,,,

$dn = "cn=".$uid.",ou=People,". $root;
#print "\$dn = $dn\n";

my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

# Bind to ldap as manager
$ldap->bind(
                        dn       => $manager,
                        password => $password,
                );

if ($exception) {
        die "Caught a signal";
} else {
        #print "\$dn = $dn\n";
        # Actually add the ldap entry
        $result = $ldap->add (
       dn    => $dn,
       attr => [ 'cn'            => $cn,
                                  'uid'           => $uid,
                 'objectClass'    => @objectclass,
                                  'uidNumber'     => $uidnumber,
                                  'gidNumber'     => $gidnumber,
                                  'loginShell'    => $loginshell,
                                  'homeDirectory' => $homedirectory,
#                                 'gecos'         => $gecos,
                               ]
        );
        $result->code && warn "failed to add entry: ", $result->error;
}
```

```
# Subroutines

my(%results);

sub finduid {
        # Find the last uid

        my($entry);
        my(@uids);
        my($lastuid);

        my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

        $ldap->bind;

        # Search for all accounts
        my($mesg) = $ldap->search( base => $root,
                                   filter => '(objectclass=account)'
                                 );

        $mesg->code && die $mesg->error;

        # loop thru all the accounts
        foreach $entry ($mesg->entries) {
                # Grab the uid name and number
        my($tmpcn) = $entry->get('uid');
        my($tmpnum) = $entry->get('uidnumber');
                # .. then stuff it into a hash
        $results{$tmpcn} = $tmpnum;
    }

        # Sort all the uids
        @uids = sort bygroup keys %results;
        # Get the last uid number and add one
        $lastuid = $results{$uids[$#uids]} + 1;

        return $lastuid;
}

sub bygroup {
        $results{$a} <=> $results{$b}
}

sub findgid {
        # Find the gid number and gid name

        my($gid) = shift;
        my($gidname, $gidnumber);
        my($entry);
        my($gidsref);
        my(@gids);

        my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

        $ldap->bind;

        # Search for all the groups
        my($mesg) = $ldap->search( base => $root,
                                   filter => '(objectclass=posixGroup)'
                                 );
```

```
        $mesg->code && die $mesg->error;

        # Loop thru all the groups
        foreach $entry ($mesg->entries) {
                # If the gid is a number..
                if ($gid =~ /\d{1,3}/) {
                        #print "\$gid is a number\n";
                        # If the gid is the same as this group's number..
                        if ($gid == $entry->get('gidnumber')->[0]) {
                                # Get the gid number and cn
                                $gidnumber = $entry->get('gidnumber')->[0];
                                $gidname = $entry->get('cn')->[0];
                                @gids = ($gidnumber, $gidname);
                                $gidsref = \@gids;
                                # and return it
                                return $gidsref;
                        }
                } else {
                #       print "\$gid is not a number\n";
                        #print $gid, $entry->get('cn'),"\n";
                        # if the gid is the same as this group's name
                        if ($gid eq $entry->get('cn')->[0]) {
                                # Get the gid number and cn
                                $gidnumber = $entry->get('gidnumber')->[0];
                                $gidname = $entry->get('cn')->[0];
                                @gids = ($gidnumber, $gidname);
                                $gidsref = \@gids;
                                # and return it
                                return $gidsref;
                        }
                }
        }
        #print "no match\n";
}

sub addgroups {
        my(@group);
        my($gidsref);
        my($groupdn);
        my($grouplist);

        if ($groups) {
                $grouplist = $groups . " " . $gidname;
        } else {
                $grouplist = $gidname;
        }

        if ($grouplist) {
                my($groupldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $

                print "\$manager = $manager, \$password = $password\n";

                $groupldap->bind(
                        dn       => $manager,
                        password => $password,
                );

                print "groups = $grouplist\n";
                @group = split(/ /, $grouplist);
```

```
                my($group);
                foreach $group (@group) {
                        $gidsref = &findgid($group);
                        #print $gidsref->[1],"\n";
                        my($tmp) = $gidsref->[1];
                        $groupdn = "cn=$tmp,ou=Group,$root";
                        print "\$groupdn = $groupdn, \$uid = $uid\n";
                        $groupldap->modify( $groupdn, add => { memberuid => $uid } )
                }
        }
}

sub checkuid {
        if ($uid !~ /[a-z]{3,8}/) {
                die "Sorry, username must consist solely of letters and be between 3
        }
        if (($uidnumber > 65535) || ($uidnumber < 0)) {
                die "Sorry, uid number must be less than 65535 and greater than 0";
        }

        my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

        $ldap->bind;

        my($mesg) = $ldap->search( base => $root,
                                   filter => '(objectclass=account)'
                                 );

        $mesg->code && die $mesg->error;

        my($entry);
        foreach $entry ($mesg->entries) {
                my($tmpcn) = $entry->get('uid')->[0];
                my($tmpuidnumber) = $entry->get('uidnumber')->[0];
                if ($tmpcn eq $uid) {
                        die "Sorry, username $uid already exists";
                }
                if ($tmpuidnumber == $uidnumber) {
                        die "Sorry, userid $uidnumber already exists";
                }
        }
}
```

ldapdeluser.pl

```
#!/usr/bin/perl -w
# ------------------------------------------------------------
# script:  ldapdeluser.pl
# Author:  Brad Marshall (bmarshal@pisoftware.com)
# Date:    20000203
#
# Purpose: Deletes a user from LDAP
#
# Copyright (c) 2000 Plugged In Software Pty Ltd.  All rights reserved.

# TODO
#   Remove the user from all groups they're in
```

```
use strict;
use Net::LDAP;
use Getopt::Std;
use Term::ReadKey;

use vars qw($opt_g);

my($uid);
my($gidnumber);
my($gidref);
my($cn);
my($dn);
my($result);
my($entry);

my($root) = "dc=pisoftware,dc=com";
my($host) = "ldap.staff.plugged.com.au";
my $exception = 0;

$SIG{'INT'} = $SIG{'QUIT'} = $SIG{'HUP'} = sub { $exception=1; };

# groupadd [-g gid [-o]] [-r] [-f] group

if (! $ARGV[0]) {
    print "$0: $0 username\n";
    exit 1;
}

if ($ARGV[0]) {
        $uid = $ARGV[0];
        if ($uid !~ /[a-z]{2,8}/) {
                die "Sorry, username must consist solely of letters and be between 3
        }
} else {
        die "Sorry, you must specify a username";
}

$dn = &finduid;

#dn: cn=support,ou=Group,dc=pisoftware,dc=com
#objectclass: posixGroup
#objectclass: top
#cn: support
#gidnumber: 140

my($manager) = "cn=Manager,$root";
print "$manager\n";

print "Please enter LDAP Managers password: ";

ReadMode 'noecho';
my $password = ReadLine 0;
chomp $password;
ReadMode 'normal';
print "\n";

print "\$dn = $dn\n";

my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";
```

```
$ldap->bind(
            dn       => $manager,
            password => $password,
        );

if ($exception) {
        die "Caught a signal";
} else {
        $result = $ldap->delete ( $dn );
        &removegroup;
        $result->code && warn "failed to delete entry: ", $result->error ;
};


# Subroutines

sub finduid {
        my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

        $ldap->bind;

        my($mesg) = $ldap->search( base => $root,
                                   filter => '(objectclass=account)'
                                 );

        $mesg->code && die $mesg->error;

        foreach $entry ($mesg->entries) {
                my($tmpuid) = $entry->get('uid')->[0];
                #print "\$tmpuid = $tmpuid\n";
                #print "$uid $tmpuid\n";
                if ($uid eq $tmpuid) {
                        #print "Found it - \$tmpcn = $tmpcn\n";
                        my($tmpdn) = $entry->dn;
                        return $tmpdn;
                }
        }
        die "Sorry, can't find the user you want to delete.";
}

sub removegroup {
        my(@groups);
        my($group);

        my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

        $ldap->bind;

        my($mesg) = $ldap->search( base => "dc=pisoftware,dc=com",
                                   filter => '(objectclass=posixGroup)');

        $mesg->code && die $mesg->error;

        foreach $entry ($mesg->entries) {
            my($tmpcn) = $entry->get('cn');
            my($tmpnum) = $entry->get('gidnumber');
            my(@members) = $entry->get('memberuid');

            #print "\@members = @members\n";
```

```
            #print "Checking $tmpcn...";

            my %t = ();
            @t{@members} = 1;
            if (exists $t{$uid}) {
                #print "yes.";
                # Push the cn onto an array
                push @groups, $tmpcn;
            }
            #print "\n";
        }

        $ldap->unbind;

        my($authldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

        $authldap->bind(
            dn       => $manager,
            password => $password,
        );

        foreach $group (@groups) {
                my($tmpdn) = "cn=$group,ou=Group,$root";
                $authldap->modify( $tmpdn, delete => { memberuid => $uid } );
        }

        $authldap->unbind;
}
```

ldapdumpusers.pl

```
#!/usr/bin/perl -w
# ------------------------------------------------------------
# script:  ldapdumpusers.pl
# Author:  Brad Marshall (bmarshal@pisoftware.com)
# Date:    20000203
#
# Purpose: Dumps the users into an /etc/passwd type file
#
# Copyright (c) 2000 Plugged In Software Pty Ltd.  All rights reserved.

use strict;
use Net::LDAP;

my($ldap) = Net::LDAP->new('ldap.staff.plugged.com.au') or die "Can't bind to ldap:

$ldap->bind;

my($mesg) = $ldap->search( base => "dc=pisoftware,dc=com",
                                        filter => '(objectclass=account)');

$mesg->code && die $mesg->error;

my($entry,$attr);

my(%results);
foreach $entry ($mesg->entries) {
        #print "DN = ",$entry->dn,"\n";
```

```
        #foreach $attr ($entry->attributes) {
        #       print $attr,": ", join(" ", $entry->get($attr)), "\n";
        #}
#       print $entry->get('cn'), " ", $entry->get('gidnumber'), "\n";
        #printf "%3d\t%s\n",$entry->get('gidnumber'), $entry->get('cn');

# postgres:!!:113:113:PostgreSQL Server:/var/lib/pgsql:/bin/sh
# uid:userpassword:uidnumber:gidnumber:gecos:homedirectory:loginshell

        my($tmpcn) = $entry->get('uid');
        my($tmpnum) = $entry->get('uidnumber');
        my($tmpgid) = $entry->get('gidnumber');
        my($tmppassword) = $entry->get('userpassword');
        if ($tmppassword) {
                $tmppassword =~ s/^{crypt}//;
        }
        #my($tmpgecos) = $entry->get('gecos') || [ "Plugged In User" ];
        my($tmpgecos) = $entry->get('cn');
        $tmpgecos ||= "";
        my($tmphome) = $entry->get('homedirectory');
        my($tmpshell) = $entry->get('loginshell');
        $tmpshell ||= "";

        #if ($tmpcn eq "root") {
        #       print "Root passwd = $tmppassword\n";
        #}
        if (! $tmppassword) {
                $tmppassword = "x";
        }
        my($tmpstr) = "$tmppassword:$tmpnum:$tmpgid:$tmpgecos:$tmphome:$tmpshell";
        #print "$tmpcn\n";
        $results{$tmpcn} = $tmpstr;
}

sub bygroup {
        (split /:/,$results{$a})[1] <=> (split /:/,$results{$b})[1]
}

my($foo);
foreach $foo (sort bygroup keys %results) {
        #printf "%3d\t%s\n",$results{$foo}, $foo;
        print "$foo:$results{$foo}\n";
}

$ldap->unbind;
```

ldaplistgroups.pl

```
#!/usr/bin/perl -w
# ------------------------------------------------------------
# script:  ldaplistgroups.pl
# Author:  Brad Marshall (bmarshal@pisoftware.com)
# Date:    20000203
#
# Purpose: Lists which groups a user is in
#
# Copyright (c) 2000 Plugged In Software Pty Ltd.  All rights reserved.
```

```
use strict;
use Net::LDAP;

my($uid);
my(@groups);
my($entry);
my($host) = 'ldap.staff.plugged.com.au';

if (! $ARGV[0]) {
    print "$0: $0 username\n";
    exit 1;
}

if ($ARGV[0]) {
    $uid = $ARGV[0];
} else {
    die "Sorry, you must specify a login";
}

my($ldap) = Net::LDAP->new($host) or die "Can't bind to ldap: $!\n";

$ldap->bind;

my($mesg) = $ldap->search( base => "dc=pisoftware,dc=com",
                                        filter => '(objectclass=posixGroup)');

$mesg->code && die $mesg->error;

foreach $entry ($mesg->entries) {
        my($tmpcn) = $entry->get('cn');
        my($tmpnum) = $entry->get('gidnumber');
        my(@members) = $entry->get('memberuid');

        #print "\@members = @members\n";
        #print "Checking $tmpcn...";

        my %t = ();
        @t{@members} = 1;
        if (exists $t{$uid}) {
                #print "yes.";
                # Push the cn onto an array
                push @groups, $tmpcn;
        }
        #print "\n";
}

$ldap->unbind;

my($groups) = join " ", @groups;
print "$uid is in $groups\n";
```

References

This tutorial is available from:
http://quark.humbug.org.au/publications/ldap_tut.html

http://quark.humbug.org.au/publications/ldap_tut.ps.gz

Understanding and Deploying LDAP Directory Services
Timothy A. Howes, Mark C. Smith and Gordon S. Good
Macmillan Network Architecture and Development Series

Implementing LDAP
Mark Wilcox
Wrox Press Ltd

Perl for System Administration
David N. Blank-Edelman
O'Reilly

The SLAPD and SLURPD Administrators Guide,
http://www.umich.edu/~dirsvcs/ldap/doc/guides/slapd/

PADL,
http://www.padl.com/

PADL's pam_ldap,
http://www.padl.com/pam_ldap.html

PADL's nss_ldap,
http://www.padl.com/nss_ldap.html

PADL's Migration scripts,
http://www.padl.com/tools.html

System Authentication Using LDAP
Brad Marshall
http://quark.humbug.org.au/publications/system_auth/sage-au/system_auth.html

---

SCO Authentication

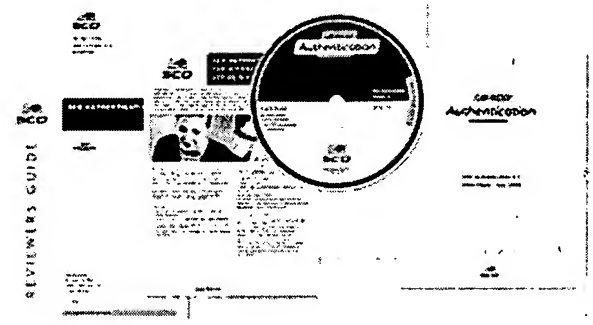
SCO Authentication

for Microsoft® Active Directory™

> REQUEST A CALL
> REGISTER
> THIRD PARTY SOLUTIONS
> SCO UPDATE

Register for a FREE evaluation kit for SCO Authentication. The evaluation kit comes with everything you need to test SCO Authentication in your Active Directory environment including; a reviewers guide, technical white paper, product brochure and 60-day evaluation software. No client-side installation needed. Register now!

SCO Authentication enables administrators to provide a secure network environment where users can utilize the same username and password for Windows, UNIX and Linux logins. System administrators no longer need to maintain password synchronizers, perform repetitive tasks in multiple systems, or spend excess time responding to user password problems.



With SCO Authentication, users only need to remember one username and password for the account that the system administrator establishes for them in Active Directory . The Active Directory account is valid for login to Windows, Linux and UNIX based systems. The system administrator is able to manage accounts from a central location using the Microsoft Management Console and UNIX command line tools.

SCO AUTHENTICATION

SCO Authentication Administration Guide

July 22, 2003

Contents

Preface

SCO Authentication allows UNIX® and Linux® users to log in and authenticate to Active Directory® in the same way that Windows® XP and Window 2000/2003 users log in and authenticate to Active Directory.

SCO AuthenticationSCO Authentication addresses the need created by having multiple operating systems and servers including Windows clients, Windows NT, UNIX, and Linux servers, and web-based services that all require users and applications to log on. SCO AuthenticationSCO Authentication integrates the authentication of users on MS Windows, UNIX, and Linux using Active Directory.

Audience Description

This guide is intended for Windows, UNIX, and Linux system administrators and system integrators who need to perform one or both of the following tasks:

- Migrate user and application authentication data from an existing UNIX Kerberos realm into Active Directory.
- Have UNIX and Linux machines that need to authenticate against Active Directory.

Conventions Used in this Guide

The following notation conventions are used throughout this guide:

- Modules, directories and filenames are bolded. For example, **/etc/pam.conf**.
- Daemon names are bolded. For example, **vascd**.
- Manual titles appear in italics. For example, *Vintela Authentication Services Installation and Configuration Guide.*
- Commands appear in a monofont. For example,

  ```
  # vastool configure pam
  ```

  Within text, commands are bolded for readability. For example,

  Using the **vastool** command line utility you can create users, delete users, and list user information.
- Variables for which you must supply a value are shown in italic monofont. For example,

  ```
  ./vastool -u matt join example.com
  ```

  Where:

  matt is a user with admin privileges in the sample network. For information on the sample network, see "Using a Sample Network" on page 10.

  example.com is the name of the Active Directory domain in the sample network.
- Menu items and buttons appear in bold. For example, click **Next**.
- Selecting a menu item is indicated as follows:

  Programs > Administrative Tools > Active Directory Users and Computers

1 Introduction

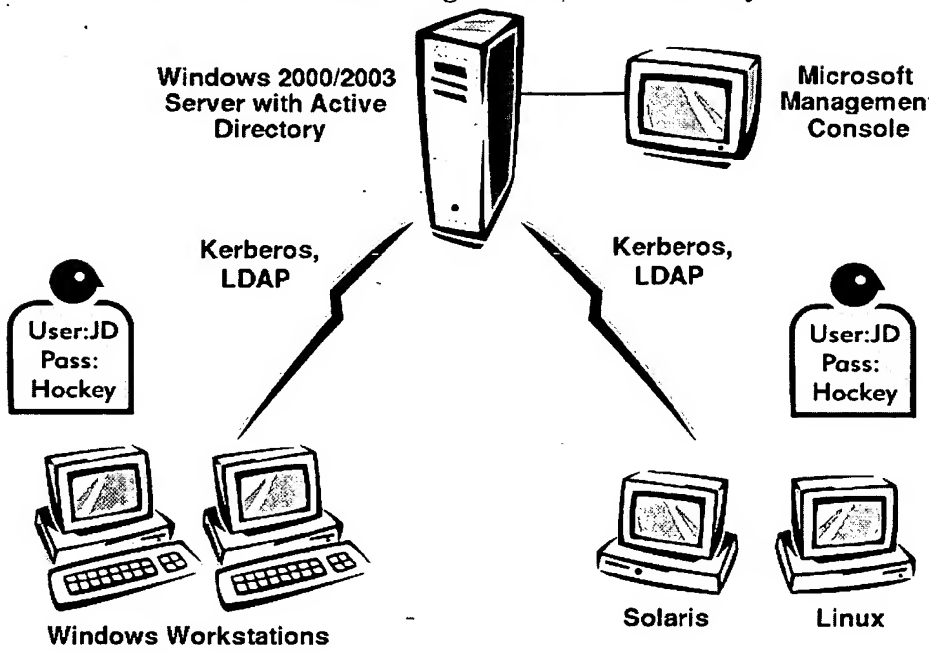
Introducing SCO Authentication

SCO Authentication allows UNIX and Linux users to log in and authenticate to Active Directory in the same way that Windows XP and Window 2000/2003 users log in and authenticate to Active Directory.

SCO Authentication provides the functionality that system administrators need to manage all user accounts in environments that use a mixture of UNIX, Linux, and Windows with Active Directory. UNIX, Linux, and Windows users have a single identity stored in Active Directory that can be administered from a single management point in the Microsoft Management Console.

The following illustrates how a user named **JD** with a password of **Hockey** logs in to Active Directory from a UNIX or Linux system. Notice that the same username and password can be used for Windows, UNIX, and Linux logins.

Figure 1. SCO Authentication Users Log in to Active Directory



SCO Authentication provides the following features and benefits:

- Fully integrated with standardized protocols supported by Windows 2000/2003, UNIX, and Linux.

- By implementing Kerberos the need for SSL configuration and key and certificate distribution is eliminated.

  SCO Authentication uses a Kerberos implementation that is compatible with Active Directory to secure all LDAP communication. Both LDAP binds and subsequent LDAP search and modify requests are fully encrypted using Kerberos based security contexts. There is no plain text or "anonymous" LDAP traffic of any kind.

- Makes efficient use of network traffic and reduces or minimizes search complexity on Windows 2000/2003 Active Directory servers.

  SCO Authentication is a scalable product that uses intelligent caching algorithms that are designed to limit the amount of network traffic and search complexity on

Windows 2000/2003 Active Directory servers. The design also makes efficient use of the UNIX host resources that make it suitable for deployment on "big iron" UNIX systems that handle hundreds of concurrent login processes.

- Provides secure user authentication even when you can't get to the network or the Active Directory server is down even on UNIX and Linux laptops.

  SCO Authentication is a robust product that is designed to work well in disconnected or loosely connected environments. For example, SCO Authentication components are suitable for use on UNIX and Linux laptops and continue to allow user authentication and UID and GID mappings even when completely disconnected from the network.

- SCO Authentication is easy to install and deploy.

  Product components can be installed and configured quickly and even automatically using the native UNIX and Linux packaging systems and intuitive command line utilities. Migration from legacy systems such as the Network Information System (NIS) or **/etc/passwd** based authentication is facilitated by a scriptable bulk user import utility within **vastool** as well as Active Directory based "NIS Map" compatibility functionality for sites that use the NIS **ypcat** utility as a distribution mechanism for more than UNIX and Linux user and Group databases.

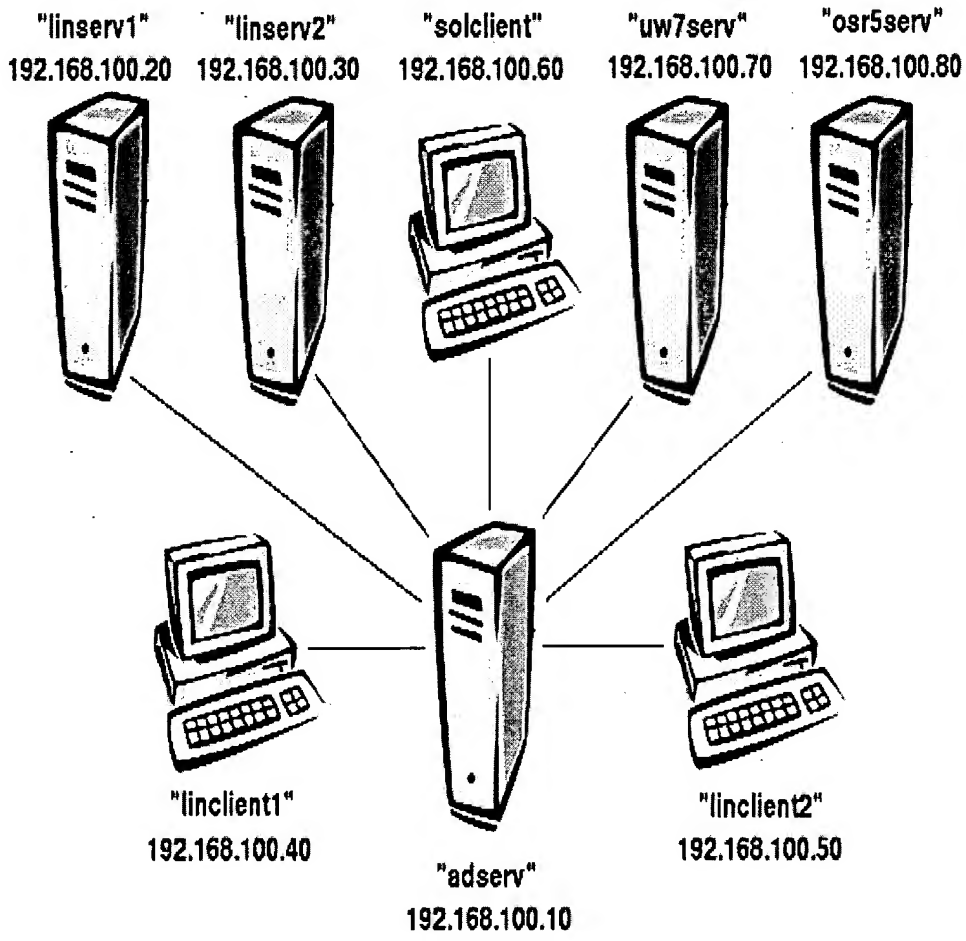
- Integrates into existing services and Open Source projects.

  SCO Authentication is a flexible product that can be customized to fit specialized user authentication requirements. Its PAM and NSS design allows it to be quickly integrated with many existing services and Open Source projects. SCO Authentication includes script-friendly command line utilities that expose its full functionality to UNIX and Linux shell programming and login scripts.

Using a Sample Network

Throughout most of this document as well as in the *SCO Authentication Installation and Configuration Guide* we have used an example scenario to assist you in your system setup. The following illustration depicts the sample network:

Figure 2. SCO Authentication Sample Network



Computers in example.com include:

Table 1. Sample System Names and Descriptions (Part 1 of 2)

	adserv. example.com	linserv1. example.com	linserv2. example.com
IP address	192.168.100.10	192.168.100.20	192.168.100.30
Operating System	Windows 2000/ 2003 Advanced Server	Red Hat Linux 8.0	SuSE 8.1
Servers/ Services	Active Directory, DNS	SSH	Apache, Samba
Realm	example.com	No Kerberos is configured or running.	No Kerberos is configured or running.
Description	Serves as the central repository for authentication data.	Remote secure login (SSH) server.	Company intranet web server, Samba server
Local User Accounts	matt, wynn, erik	wynn	matt

Table 2. Sample System Names and Descriptions (Part 2 of 2)

	linclient1. example.com	linclient2. example.com	solclient. example.com
IP address	192.168.100.40	192.168.100.50	192.168.100.60
Operating System	UnitedLinux 1.0	UnitedLinux 1.0	Solaris 8
Servers/ Services	None	None	None
Realm	No Kerberos is configured or running.	No Kerberos is configured or running.	No Kerberos is configured or running.

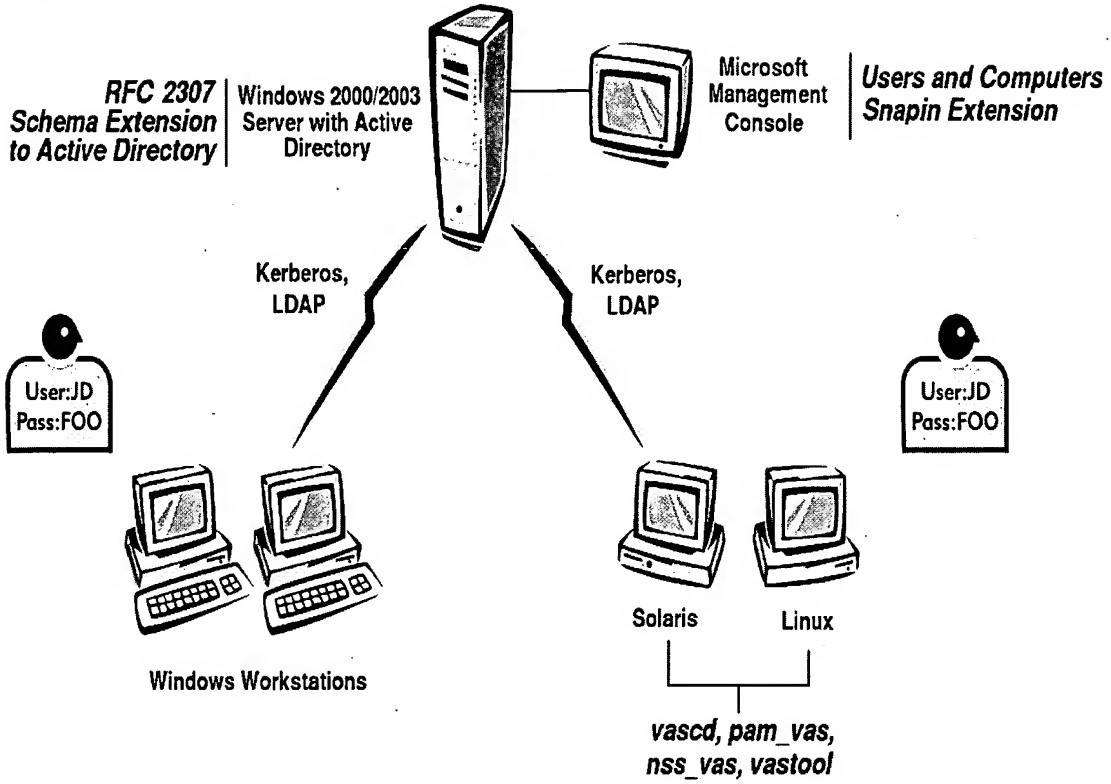
	linclient1. example.com	linclient2. example.com	solclient. example.com
Description	Linux workstation.	Linux workstation.	Solaris workstation
Local User Accounts	matt	wynn	erik

2 Introduction to SCO Authentication Components

This section provides a brief description of the main software components that are installed with the SCO Authentication product. They are as follows:

A graphic depiction of SCO Authentication components and where they reside follows:

Figure 3. SCO Authentication Components



Schema Extension Utility

The Schema Extension Utility is a simple application that allows the administrator to apply the RFC 2307 schema extensions for LDAP management of UNIX account information. Install the Schema Extension Utility by selecting the **Schema Master** installation profile on the Windows 2000/2003 SCO Authentication installer. Once the Schema Extension Utility is installed run it using the instructions provided in the *SCO Authentication Installation and Configuration Guide.*

Users and Computers Snapin Extension

The Users and Computers Snapin extension within the Microsoft Management Console adds the **UNIX Account** tabs to the Users and Groups properties dialog. The extension is installed by the SCO Authentication installer on the Master PDC as part of both the **Admin Workstation** and **Schema Master** installation profiles.

The vascd Daemon

vascd is a daemon that provides a local proxy for Active Directory and locally caches user and group account information from the Active Directory server. **vascd** must be started on UNIX and Linux workstations in order for SCO Authentication to operate correctly. When started, **vascd** authenticates to Active Directory using credentials that were established at the time that the computer object was created in the Active Directory domain. (For more information on computer objects see, "Computer Objects" on page 23 and the *SCO Authentication Installation and Configuration Guide* for more information). **vascd** then uses this secure connection to Active Directory to proxy and cache user and group account information for other processes.

The use of **vascd** provides several important features:

Security -Because of the way that PAM and NSS subsystems operate, most LDAP-based UNIX account management solutions require that anonymous or public access to UNIX account properties be allowed. Since **vascd** authenticates as an Active Directory domain computer, **vascd** can access UNIX account information that is protected by Active Directory access control restrictions.

Scalability - Also, because of the way that PAM and NSS subsystems operate, most LDAP and NIS-based UNIX account management solutions generate excessive numbers of LDAP connections and LDAP search requests. This results in dramatically increased network traffic and load on the LDAP server. **vascd** establishes a single connection that is used to proxy all information requests for all processes. At the same time, **vascd** is able to perform intelligent caching of frequently used information so that LDAP traffic is reduced to the absolute minimum.

Disconnected Operation - **vascd** maintains a persistent cache of frequently used information. This makes it possible for the entire SCO Authentication system to continue to operate in environments where the network connection to the Active Directory server is unreliable or completely unavailable. This is particularly useful for dialup and laptop users.

For additional information on **vascd**, see the **vascd** man page.

The pam_vas Module

The pluggable authentication module (PAM) library is used by applications that need to authenticate usernames and passwords. System administrators can configure how users are authenticated to the UNIX or Linux host by configuring each step of the authentication process and by choosing which PAM module to use for each of the steps. The **pam_vas** module allows login applications to authenticate usernames and passwords against Active Directory using the Kerberos protocol.

Using **pam_vas** provides the following features:

Disconnected Authentication - **pam_vas** continues to allow Active Directory logins when UNIX and Linux workstations are disconnected from the network or when the Active Directory server is not available.

Automatic Home Directory Creation - Administrators can configure the **pam_vas** module to automatically create users' home directories if they do not exist at login time. The home directory is set up with the proper ownerships and permissions and is populated with the information stored in **/etc/skel**.

UID Conflict Checking - When storing UNIX account information in an Active Directory repository, it is easy to create UID conflicts with local system accounts stored in **/etc/passwd**. Duplication of UIDs between Active Directory and **/etc/passwd** can create a security hole where a local system user with the same UID as an Active Directory user could access that Active Directory user's files, and vice versa. **pam_vas** prevents this by not allowing Active Directory users to log in if they have a UID conflict and their UID is greater than 1000.

Machine Based Access Control - pam_vas allows you to selectively control which Active Directory users can interactively log on to a certain machine. You can configure a **users.allow** and a **users.deny** file to deny or allow local access to certain SCO Authentication users or groups.

Password Administration - pam_vas allows users to change passwords that are stored in Active Directory. This allows users to use one password on all the systems where the SCO Authentication client is running.

For information on configuring the PAM module, see the "PAM Configuration" on page 39 as well as the **pam_vas** man page.

The nss_vas Module

nss_vas is the SCO Authentication Name Service Switch (NSS) module for UNIX and Linux NSS subsystems. The NSS subsystem is used by applications to obtain UNIX account information such as UID, GID, home directory, and default login shell. The addition of the **nss_vas** module allows UNIX account information to be pulled from Active Directory using the LDAP and Kerberos protocols.

Unlike other LDAP-based NSS modules, the **nss_vas** module does not communicate directly with Active Directory. Instead, **nss_vas** contacts the **vascd** daemon running on the same system. **vascd** is then responsible for either satisfying the NSS information request using its persistent cache or for establishing a *secure* LDAP connection with Active Directory. Using **vascd** as a proxy for NSS information allows **nss_vas** to operate much more securely and efficiently than other LDAP-based NSS modules.

The vastool Command Line Utility

vastool is a command line utility that provides commands to configure the SCO Authentication components, access information in Active Directory, and store information in Active Directory. It is designed for ease-of-use in scripts and cron jobs. It also

provides migration tools for NIS and local user account databases.

For more information on **vastool** and each individual command, see the **vastool** man page. Certain **vastool** commands are referenced throughout this guide.

The following table lists **vastool** commands and functionality.

Table 3. vastool Commands

Command	Function
attrs	Lists an Active Directory object's attributes.
configure	Updates configuration files to use the SCO Authentication commands.
create	Creates users, groups, and computer objects in Active Directory.
delete	Deletes users, groups, computer objects, and NIS Map objects in Active Directory.
flush	Flushes cached client daemon information.
group	Modifies group membership.
join	Joins the computer to the domain.
kinit	Performs kinit functions and obtains Kerberos tickets.
klist	Performs klist functions and shows the Kerberos ticket cache.
kdestroy	Performs kdestroy functions and destroys Kerberos tickets.
license	Installs your user license.
list	Lists users and groups in Active Directory.
load	Loads and creates users and groups in Active Directory.
nis-import	Loads NIS Maps into the directory.

Command	Function
passwd	Changes a user's or your own password.
realms	Detects the realms on your network and the servers providing LDAP and Kerberos services for those realms.
timesync	Synchronizes the system clock with an SNTP server.
unconfigure	Reverts SCO Authentication configuration changes.
unjoin	Removes the local computer from the domain.
ypcat	Provides functionality similar to the NIS ypcat utility.

For more information on **vastool** and each command, see the **vastool** man page.

3 Managing UNIX and Linux Systems in Active Directory

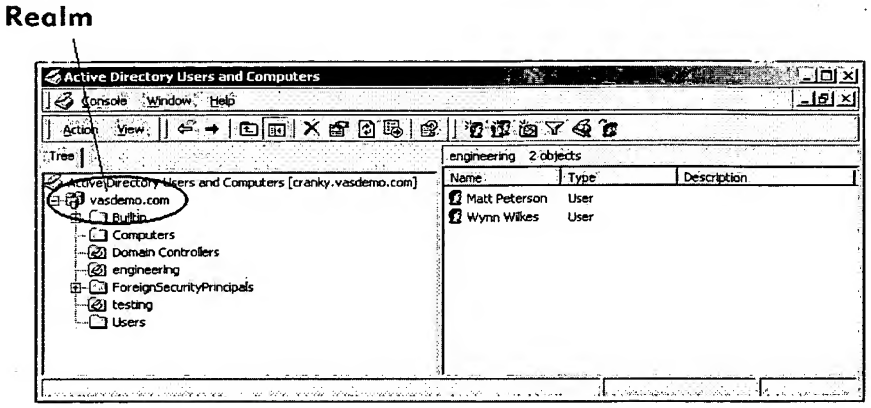
This section addresses system configuration topics that are related to SCO Authentication. These topics include the following:

The Kerberos Realm and KDC

After installing the SCO Authentication components on UNIX and Linux systems, the first configuration step is to configure the Kerberos realm and the key distribution centers (KDCs) that are used to obtain Kerberos authentication tickets for users, services, and computers. When using Active Directory, it is important to note that the Kerberos Realm is the same as the Active Directory domain and that a KDC is an Active Directory server. Figure 4 on page 22 shows where the Realm and KDC values are displayed in the Users and Computers Snapin.

Before attempting to install and configure SCO Authentication components on UNIX and Linux systems you should obtain values for the Kerberos realm. To determine the correct realm, open the Users and Computers Snapin and record the values as illustrated in the following:

Figure 4. Realm displayed in the Users and Computers Snapin



Configuration for the realm and KDC settings is saved in the **/etc/opt/vas/vas.conf** file. However, it is recommended that system administrators use the **vastool join** command to set and change the realm and KDC settings. See the **vastool** man page for complete **vastool** command documentation.

Computer Objects

In order to securely communicate with Active Directory it is necessary to create and maintain a computer object in Active Directory for every UNIX and Linux system that uses SCO Authentication to authenticate users. This section provides additional details beyond what is outlined in the *SCO Authentication Installation and Configuration Guide.*

Creating Computer Objects

Computer objects are created using the **vastool create** command or as part of the **vastool join** process which additionally configures the realm, NSS, and PAM. **vastool create** offers additional flexibility over **vastool join** in that it only performs the step of computer creation. **vastool create** also offers command line options that allow computer objects to be created outside the default computer container.

When creating computer objects it is important to remember that when using **host/** as the object name **vastool** looks up the current host name of the system on which **vastool** is being run and uses this name as the name of the computer object being created. In environments where all UNIX and Linux hosts have been assigned unique host names this is not a problem. However, in environments where UNIX and Linux systems are not assigned the same unique hostname each time they boot then administrators should create computer objects using an explicit name, for example **host/lab12**. As part of the computer creation process, a randomly generated password for the computer is saved to **/etc/opt/vas/host.keytab**. If **host.keytab** is deleted or corrupted, then **vascd** will not be able to authenticate as the computer object. If **host.keytab** is compromised by unauthorized root access, then the password for this computer should be assumed to be compromised as well. You can reset the computer object's password by running **vastool create** which generates a new password for the computer object if the computer object already exists. This is useful in imaged environments where UNIX and Linux hosts are frequently re-installed. You can also completely delete the computer object, and then recreate it. For information on deleting objects, see "Deleting Computer Objects" on page 24.

vastool examples follow:

```
$ vastool -u matt create -c "ou=eng,dc=example,dc=com" host/
```

Authenticates as the user *matt*, uses the current hostname as the computer name and creates the computer object in the *ou=eng,dc=example,dc=com* container.

```
$ vastool -u matt create host/linclient1.example.com
```

Authenticates as the user *matt*, using *linclient1.example.com* as the computer name and creates the computer object in the default computers container.

Administrative Privileges

The process of creating a computer object requires administrative privileges. In other words, the user specified with the **-u** option must be a member of the Windows 2000/ 2003 "Domain Admins" security group. Because of various security subtleties, the "Administrator" user as such *can not* be used for performing SCO Authentication administration tasks. The ability to create computers can be delegated to non-administrative users. For instructions, see "Delegating Administrative Privileges for Computer Creation" on page 25.

Moving Computer Objects

To move computer objects from one organizational unit to another use the "drag-n-drop" functionality of the Users and Computers snapin. Other than NIS map considerations explained in Chapter 7, "NIS Migration," on page 87 there are no additional administrative tasks associated with moving computer objects.

Deleting Computer Objects

Computer objects should be deleted when a UNIX or Linux host is removed from service, the system has to be re-installed with a different operating system, or if a system has had a security breach (for example, unauthorized root access).

To delete unused computer objects, use the Users and Computers Snapin or use the **vastool delete** command or the **vastool unjoin** process which deletes the computer object and unconfigures PAM and NSS. The **vastool create** command resets a computer object's password when the computer object already exists.

To delete the computer object before re-creating it, run the following as **root** (in environments where the hostname is the name of the computer in Active Directory):

```
# vastool -u matt delete host/
```

If you specified a name for the computer object, then as **root** use the command:

```
# vastool -u matt delete host/linclient1.example.com
```

Now run the **vastool create** command.

Security Considerations

The default permissions for a computer object restrict the computer from accessing sensitive data in Active Directory. The schema extensions are carefully designed to allow computers with default permissions to access only the UNIX account data that is absolutely necessary for the normal operation of the **vascd** daemon. We recommend that administrators not modify the default permissions for the computer object to make them either more or less restrictive. Changing the computer object permissions could disrupt normal operation or create a security liability that might result in the compromise of sensitive data.

Delegating Administrative Privileges for Computer Creation

It is often necessary to allow trusted users to perform tasks that are normally reserved for system administrators. Delegating trust to selected users can not only ease the pain of system administration but can also increase efficiency within your IT organization. One person adding computer objects to Active Directory one at a time is at the very

least a cumbersome task; if the number of computers to be added is up in the hundreds, the task changes from being cumbersome to being prohibitively time consuming. In a situation like this, the need to delegate computer creation becomes essential.

Maintaining tight control over administrator privileges is another important IT issue. Trusted users could be added to the Domain Admin group and in the process obtain sweeping domain administration privileges, but wide open access to an Active Directory domain -- even to trusted users --poses potential security risks that are best avoided. Granting specific authorizations to specific users while holding back on other authorizations helps maintain control over who is doing what to your Active Directory data.

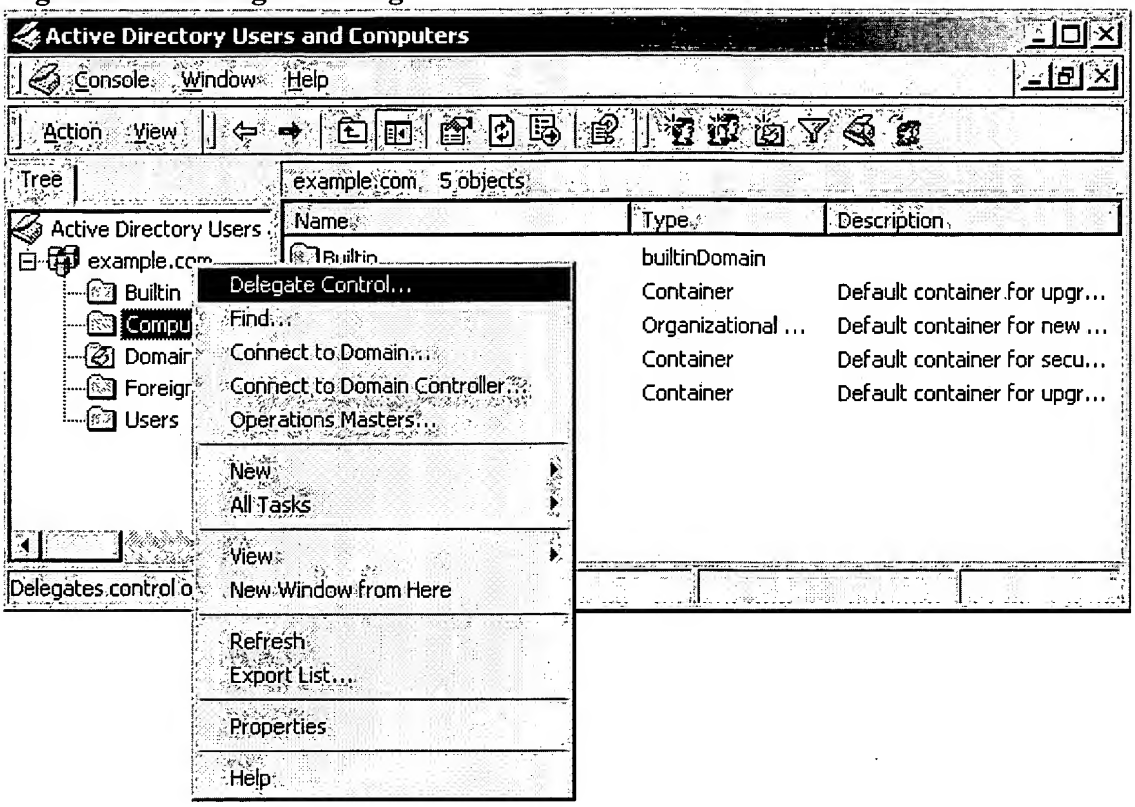
To set up users and/or groups with computer creation authorization, perform the following steps:

1. Select **Start > Programs > Administrative Tools > Active Directory Users and Computers**.

2. Right-click on the Active Directory domain name in the **Active Directory Users and Computers** tree on the left.

   The following appears:

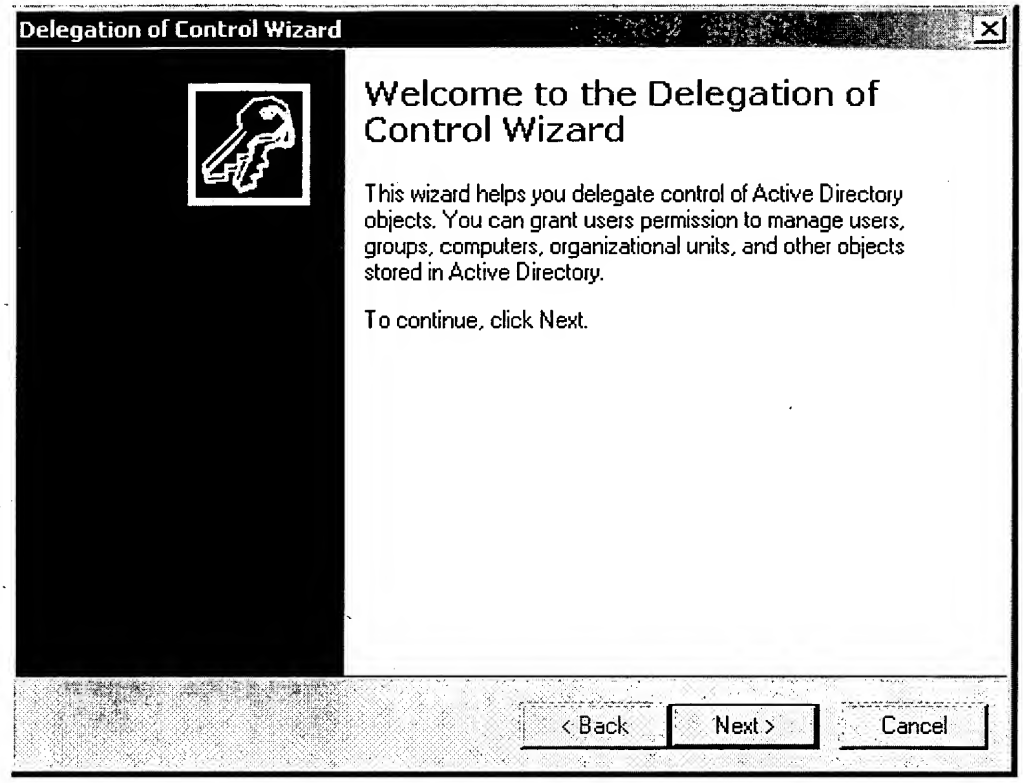
Figure 5. Initiating the Delegation of Control Wizard



3. Click **Delegate Control** to start the Delegation of Control Wizard.

   The following appears:

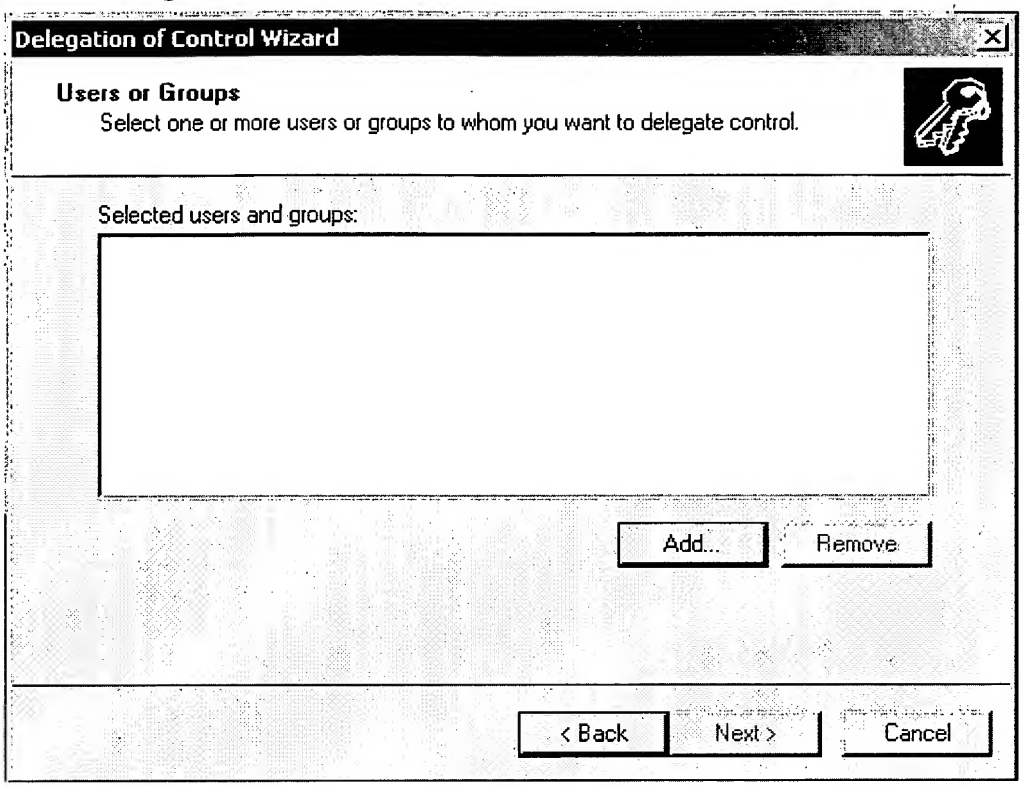
Figure 6. Welcome Screen



4. Click **Next**.

   The following appears:

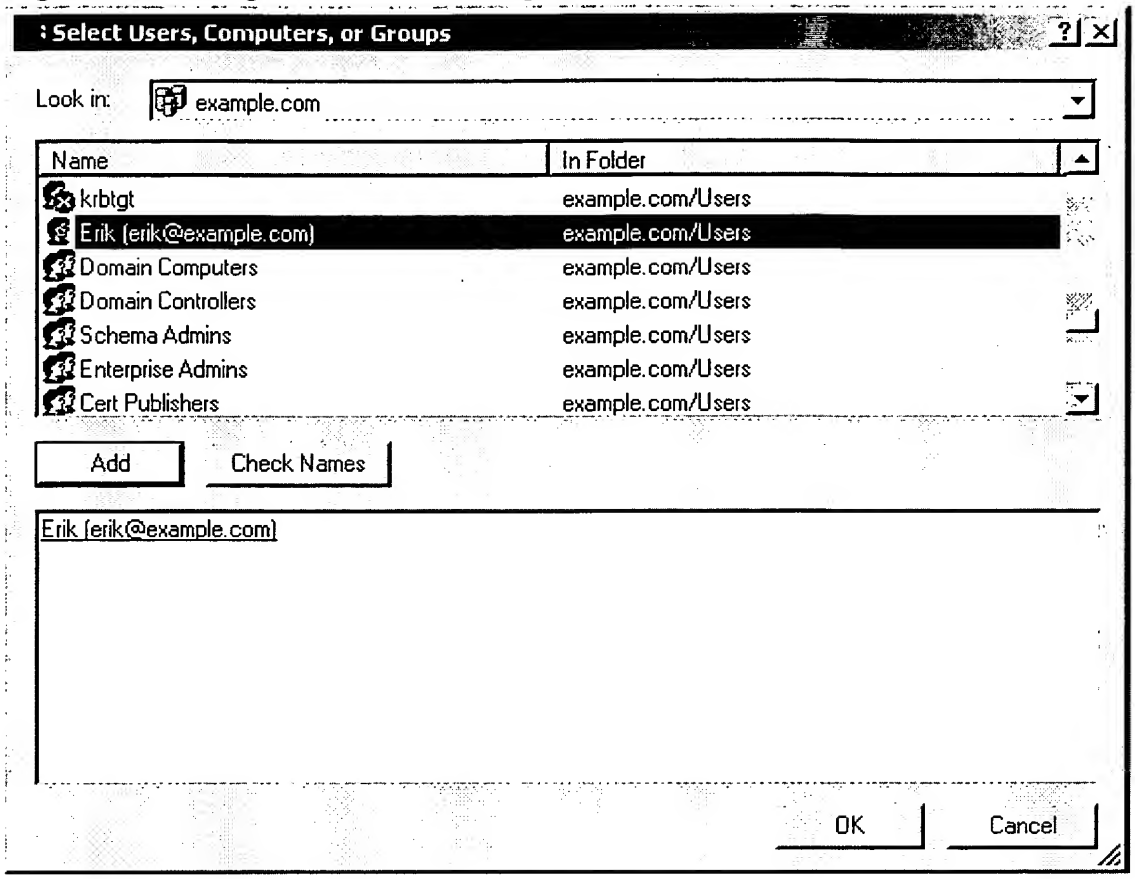
Figure 7. Adding Users or Groups



5. Click **Add**.

   The following appears:

Figure 8. Selecting Users and/or Groups to Add

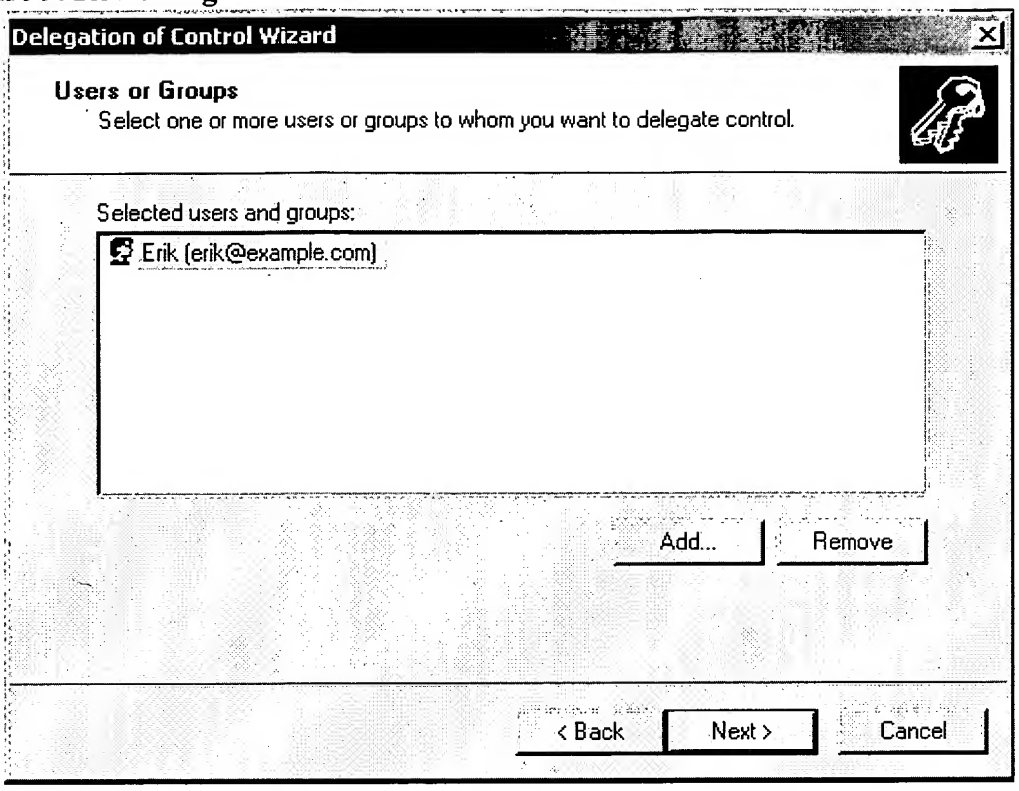


6. Select the users and/or groups to whom you want to delegate the computer creation authorization and then click **Add**.

   Note: If you are setting up groups with computer creation authorization, be sure to assign trusted users to the groups.

7. Click **OK** when finished to return to the **Users or Groups** selection window. Your selected users appear in the window as follows:

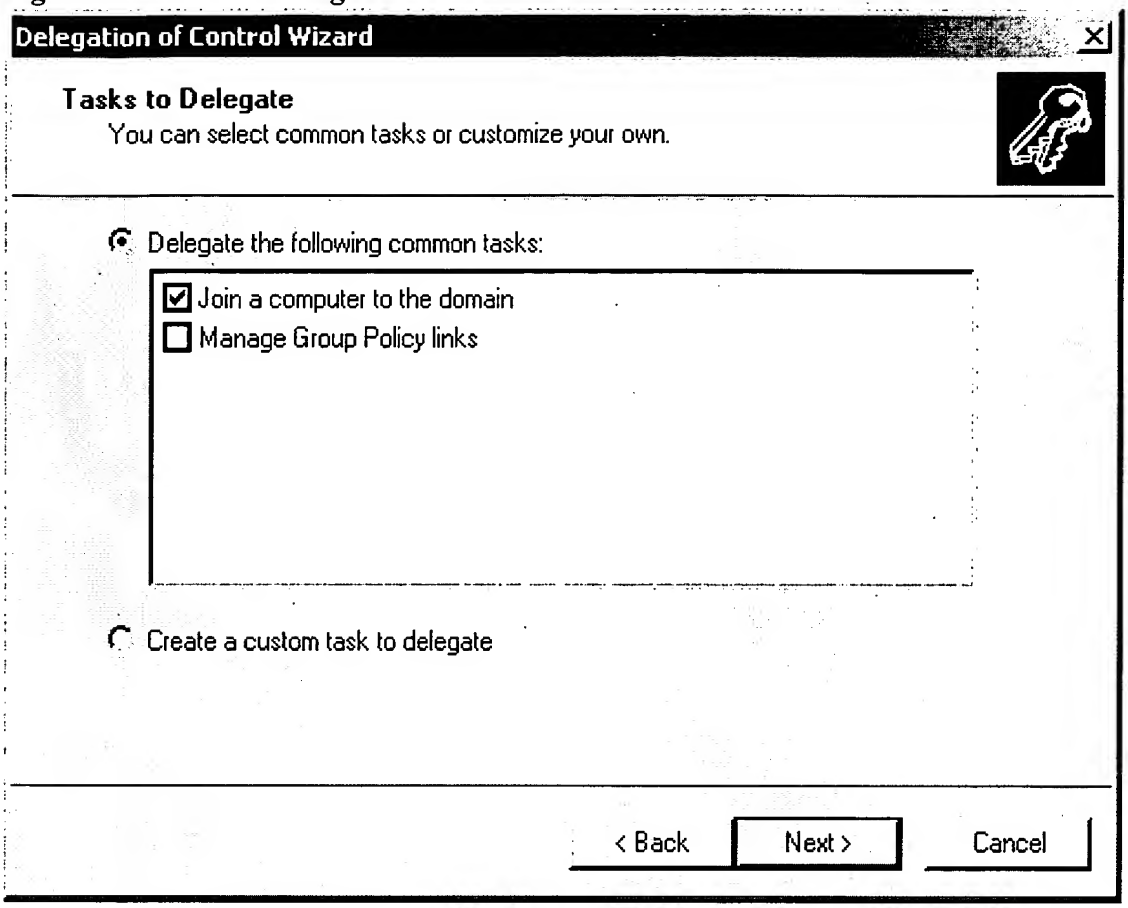
Figure 9. Reviewing Additions



8. Click **Next.**

   The **Tasks and Delegate** window appears.

Figure 10. Tasks to Delegate

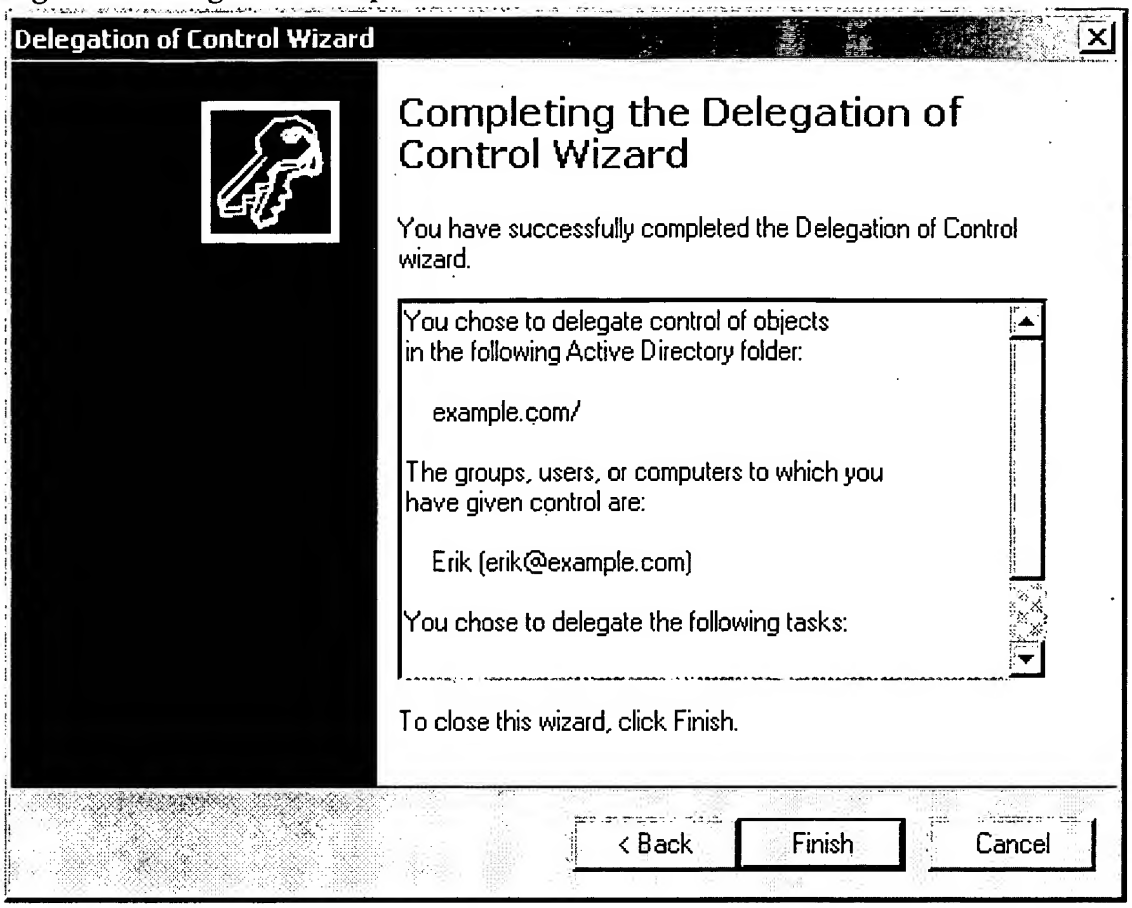


9. Click in the checkbox next to **Join a computer to the domain** and then click **Next**.

   The Completing the Delegation of Control window appears

10. If the information is correct, click **Finish**.

Figure 11. Delegation Complete



Managing Different Hostnames and Active Directory Domain Names

Most networks that utilize Active Directory use the same domain name for both their Internet and Active Directory domains. When this happens all of the computers on the network are already set up for proper Active Directory interaction in DNS on UNIX and Linux systems.

By default, the name of the Active Directory computer object that is created for a SCO

Authentication-enabled computer is identical to that computer's hostname. So, if a SCO Authentication-enabled computer named **solclient** joins the Active Directory domain, an associated computer object is named **solclient** is created within Active Directory. If **example.com** is the name of the Active Directory domain, Active Directory must be able to resolve **solclient.example.com** (*<computer object>.<AD domain>*) in DNS.

However, if the Active Directory domain name does not match the Internet domain name of the computer, one of the following must be performed:

- Add new hostname entries in DNS that map each SCO Authentication-enabled computer into the Active Directory domain.
- Specify the name of the computer's Active Directory computer object explicitly using **vastool**.

These methods are discussed in the following sections.

Adding New Hostname Entries

Assume your Internet domain name is **example.com** and your Active Directory domain name is **sample.com**. Also, assume you have a set of five computers in the **example.com** domain that authenticate against Active Directory using SCO Authentication.

Before these computers can authenticate against Active Directory you must add the Active Directory domain name for each computer to DNS for each system that maps into the Active Directory domain. Since the computers must still be resolvable as **sample.com** both entries must appear in DNS.

Table 4. DNS Entries

Internet Domain Names	Active Directory Domain Names	DNS entries
fred.example.com	fred.sample.com	fred.sample.com fred.example.com

Internet Domain Names	Active Directory Domain Names	DNS entries
barney.example.com	barney.sample.com	barney.sample.com barney.example.com
wilma.example.com	wilma.sample.com	wilma.sample.com wilma.example.com
betty.example.com	betty.sample.com	betty.sample.com betty.example.com
dino.example.com	dino.sample.com	dino.sample.com dino.example.com

Using vastool to Specify a Computer's Active Directory Object

Assume your Internet domain name is **example.com** and your Active Directory domain name is **sample.com**. Also assume you have a set of five computers on the **example.com** domain that need to authenticate against Active Directory using SCO Authentication. They are as follows:

fred.example.com

barney.example.com

wilma.example.com

betty.example.com

dino.example.com

If the only DNS entry for **barney** uses **example.com** as the domain name, then Active Directory won't find the **barney** computer object. This is because Active Directory looks for **barney.sample.com** which is not in DNS. In order for **vastool join** to work with **barney**, the name of the computer object must be specified on the **vastool** com-

mand line using the **-n** parameter as follows:

```
vastool -u adminuser join -n barney sample.org
```

By specifying the computer object on the **vastool** command line, Active Directory won't need to consult DNS to figure out which computer object in Active Directory should be used.

Running vascd

The **vascd** daemon (or process) must be started on UNIX and Linux workstations in order for SCO Authentication to operate correctly. **vascd** plays an important role in providing many of the security, scalability, and stability features of SCO Authentication. For more information on **vascd**, see, "The vascd Daemon" on page 15. When started, **vascd** authenticates as the computer using the credentials that were established at the time that the computer object was created in the Active Directory domain. For information on computer objects, see "Computer Objects" on page 23. After initialization, **vascd** acts as a proxy for requests for information that is stored in Active Directory. **vascd** then provides the requested information either from a cache, or by contacting the Active Directory server.

The vascd Data Cache

In order to minimize network traffic and the load on the Active Directory server, **vascd** aggressively caches data that is retrieved from the directory server so that subsequent requests can be satisfied from the cache without having to contact the Active Directory server. The cache is only updated when it is determined that a change has been made in the directory or when a new user logs in. Nevertheless, because of the way that the NSS subsystem is called it is not uncommon for hundreds of requests to be generated in a matter of seconds. Therefore, in order to further reduce the load on the network and on the directory, **vascd** enforces a blackout period during which all NSS-initiated requests are resolved from the cache. By default, the blackout period is set to a value of 10 minutes.

This means that changes to UNIX Account information take up to 10 minutes by default, to become visible to SCO Authentication clients. New users can immediately log in to SCO Authentication clients without changing the blackout period. For environments where changes must take affect faster, change the blackout period by changing a setting in **/etc/opt/vas/vas.conf**. In the [vascd] section, add a parameter for "update-interval" as follows:

```
[vascd]
  update-interval = 300
```

This specifies the length of the blackout period in seconds. The default is 600 seconds. For more information on the blackout period, see the **vascd** man page. Whether this results in additional network traffic depends on the number of UNIX and Linux hosts and their use. In small installations (less than 100 hosts or less than 100 users) the blackout period can be safely reduced. In larger installations it is recommended that the blackout period be left at the default value or increased to 30 minutes or 1 hour. Regardless of the blackout period, the administrator can force **vascd** to update the cache immediately by signaling **vascd** with SIGHUP or by executing **vastool flush**.

Using vascd with nscd

The Name Service Caching Daemon (**nscd**) is a simple and generic caching daemon that is started by default on some UNIX and Linux operating systems to reduce system and network load due to repeated calls to the NSS subsystem. On systems where **nscd** is installed, **nscd** caching of passwd and group data is disabled as part of the **vastool join** command. This is done because SCO Authentication has its own caching mechanisms and disabling **nscd** caching of group and password information saves system resources. However, it is possible to re-enable **nscd** caching of passwd and group databases. You might want to do this if you're using NIS with SCO Authentication as part of your migration strategy.

We recommend that after you install a new version of **nscd** (or patch kits that might affect the **nscd** configuration) that system administrators examine the **/etc/nscd.conf** file to ensure that passwd and group caching is disabled. Additionally, if User or Group account changes made in Active Directory do not appear to be in effect on a UNIX or Linux machine, one troubleshooting step (in addition to checking the **vascd**

blackout period) should be to check the **/etc/nscd.conf** file to ensure that passwd and group caching is disabled.

Disconnected Mode

When **vascd** is unable to contact the KDC or the Active Directory server, it reverts to a disconnected mode. While in this mode all NSS and PAM requests are resolved from the cache. Disconnected mode can be entered for one or more of the following reasons:

- The computer object is deleted. If the UNIX or Linux system's computer object is deleted then **vascd** can no longer communicate with Active Directory because it can not authenticate. The solution to this problem is to re-create the computer object, then restart **vascd**.

- The **/etc/opt/vas/host.keytab** file is invalid. If **/etc/opt/vas/host.keytab** is deleted or becomes corrupt then **vascd** is no longer able to communicate with Active Directory because it does not have credentials to authenticate as the computer. The solution to this problem is to delete then re-create the computer and restart **vascd**.

- The Active Directory server is down.

- The computer is physically disconnected from the network or the network is down.

  Since most UNIX account information requests are correctly handled by the cache, it is often difficult to tell if **vascd** is in disconnected mode. One sure way to determine the state of **vascd** is to look at the system log file. **vascd** makes a short log entry each time the connection mode changes.

Finally, the **vascd** disconnected mode is not intended to solve all problems related to completely disconnected situations. For example, **vascd** does not have any control over the ability of the system to continue to resolve DNS names or to continue accessing network file systems in an abrupt and completely disconnected situation.

PAM Configuration

PAM is an API that allows applications' authentication mechanisms to be extended by system administrators without having to modify the applications themselves.

pam.conf

PAM enabled services are configured through either the **/etc/pam.conf** file, or by individual files in the **/etc/pam.d/** directory. The **/etc/pam.d** directory approach is used by newer versions of PAM found in most Linux distributions such as UnitedLinux, Red Hat, and SuSE. Solaris uses the **/etc/pam.conf** approach. Both approaches use configuration files that have similar syntax.

There are four different types of entries in the PAM configuration file:

- auth - checks username/password pairs.
- account - handles processing of account restrictions.
- password - handles password changing
- session - handles any initialization that needs to occur for an interactive login session.

PAM allows you to configure a service to use a "stack" of PAM modules. This means there is a list of PAM modules that are consulted to perform the various authentication tasks. The **pam_vas** module should be configured to be the first module in the stack. **pam_vas** returns an IGNORE result when the user being authenticated is not an Active Directory user. This allows the other standard PAM modules to be called to authenticate local **(/etc/passwd)** users.

Refer to your system administration documentation for your operating system concerning PAM to make sure you understand PAM before performing any customizations beyond the changes made by the **vastool configure** commands described in "Configuring PAM with vastool" .

Configuring PAM with vastool

You can configure individual services to use the **pam_vas** module using **vastool configure pam**. For example, to configure the **ssh** daemon to use the **pam_vas** module, run the following command as **root**:

```
# vastool configure pam sshd
```

To configure all services to use **pam_vas**, do not specify a service name. The following command configures all services to use **pam_vas**:

```
# vastool configure pam
```

The following is an example of what the PAM configuration file for **sshd** should look like after running **vastool configure pam sshd**:

```
auth  required   /lib/security/pam_env.so

auth  [ignore=ignore success=done default=die] \

/opt/vas/lib/security/pam_vas.so get_tgt create_homedir

auth  sufficient /lib/security/pam_unix.so likeauth nullok

auth  required   /lib/security/pam_deny.so

account  [ignore=ignore success=done default=die] \
/opt/vas/lib/security/pam_vas.so

account     required      /lib/security/pam_unix.so

password  [ignore=ignore success=done default=die] \
/opt/vas/lib/security/pam_vas.so

password  required    /lib/security/pam_cracklib.so retry=3 type=

password  sufficient  /lib/security/pam_unix.so nullok use_authtok \
md5 shadow

password  required    /lib/security/pam_deny.so
```

```
session     required      /lib/security/pam_limits.so

session  [ignore=ignore success=done default=die] \

/opt/vas/lib/security/pam_vas.so

session     required      /lib/security/pam_unix.so
```

The entries for **sshd** on a system that uses **/etc/pam.conf** would look the same, except each line would be prefixed with **sshd**.

Reverting PAM Configuration Changes

You can remove the **pam_vas** module from any service's PAM configuration using the **vastool unconfigure pam** command. For example, to unconfigure the **ssh** daemon from using the **pam_vas** module, you would run the following command as **root**:

```
# vastool unconfigure pam sshd
```

To revert the configuration for all services, do not specify a service name. The following command removes **pam_vas** from the configuration of all services:

```
# vastool unconfigure pam
```

Kerberos Ticket Caches

The **pam_vas** module uses the Kerberos protocol to authenticate users against Active Directory. The Kerberos protocol allows users to obtain one "ticket" using their password that they can then use to obtain other "tickets" to authenticate to services such as the Active Directory LDAP server. This provides a single sign on mechanism that does not make users repeatedly enter in their password.

By default, when a user establishes a login session that used the **pam_vas** module, they have a ticket cache stored in their home directory with the name **.krb5cc**. The KRB5CCNAME environment variable is also set to allow other Kerberos-aware applications to locate the ticket cache.

These tickets do not contain the users' password, but they could be used in a brute force attack to determine passwords, much like an attacker could use the password hashes in **/etc/shadow** if those were publicly available. Since the file **$HOME/.krb5cc** is sensitive, it is created with permissions of 0600 (readable and writable only by the file owner) and owned by the user. This file should not be moved or modified by the user. This file is cleared out when the login session ends.

You can examine what tickets you have using the **vastool klist** command. This lists the tickets you have, as well as their expiration times.

If the **pam_vas** module cannot securely create the credentials cache file in a user's home directory, it uses a default location of **/tmp/krb5cc_xxx** where **xxx** is the user's UID, with the same permissions of 0600.

User Home Directory Creation

By default **pam_vas** creates users' home directories if they do not exist. The directories are created with the appropriate permissions of 0700 (readable, writable, and executable only by the owner of the directory) and owned by the user. The files in **/etc/skel** are also copied into the new home directory.

This behavior can be disabled by removing the **create_homedir** option from the auth entries. You might want to do this in environments where home directories are served over the network through distributed files systems.

To disable automatic home directory creation, modify the following line:

```
auth  [ignore=ignore success=done default=die] \
/opt/vas/lib/security/pam_vas.so get_tgt create_homedir
```

to look like:

```
auth  [ignore=ignore success=done default=die] \
/opt/vas/lib/security/pam_vas.so get_tgt
```

Using pam_vas with Non-Shell Login Services

You can use **pam_vas** with services that do not provide shell login functionality, such as Apache or a mail server. **pam_vas** has some options to make the authentication method more efficient in this case. For example, you can disable the obtaining of a Kerberos Ticket Granting Ticket (TGT) by removing the **get_tgt** option from the auth entry for **pam_vas**. This reduces the amount of Kerberos network traffic that has to be performed for each login, and does not store any Kerberos tickets on disk.

For these types of non-shell login services, disable the creation of users' home directories as well. These customizations can be done by modifying the following line:

```
auth  [ignore=ignore success=done default=die] \
/opt/vas/lib/security/pam_vas.so get_tgt create_homedir
```

to look like:

```
auth  [ignore=ignore success=done default=die] \
/opt/vas/lib/security/pam_vas.so
```

Debugging PAM Problems

If you experience problems with the **pam_vas** option, you can enable verbose debugging for the **pam_vas** module using the debug option. The debug option can be appended to any **pam_vas** entry. The debug output goes to the authentication system log via syslog. On Red Hat, these log messages are logged to **/var/log/secure** by default. On UnitedLinux based distributions, these messages are logged to **/var/log/messages** by default. On Solaris, there is no default syslog setup for auth messages. To enable syslog for auth messages, add the following lines to **/etc/syslog.conf**, and then restart **syslogd**. Make sure that you use tabs to separate the left columns from the right columns.

```
auth.alert<TAB>/dev/console
auth.crit<TAB>root
auth.info;auth.debug<TAB>/var/log/auth
```

pam_vas and Account Restrictions

Active Directory allows you to set several restrictions on user passwords, such as expiration dates and policies that force the user to change their password at next login. The **pam_vas** module also supports these options.

When users log in and their password is expired, **pam_vas** prompts for the old password, and a new password. However, not all services that use the PAM interface support this process. For example, KDE's **kdm** does not process the PAM requests from **pam_vas** correctly, but GNOME's **gdm** and a system login prompt do. The Solaris **dtlogin** also processes the requests correctly. sshd by default does not process these PAM requests, but can be configured to do so. For information on configuring sshd, see"Using SCO Authentication with SSH" on page 76.

Disconnected Authentication

pam_vas allows users to continue to use their Active Directory passwords to authenticate even when the network connection to the Active Directory server is disrupted. Users must login at least once while in the connected state in order to be able to log in when the system is disconnected.

Disconnected authentication operates by caching user names and a SHA-1 hash of their password when they login during a normal connected authentication operation. When the system is disconnected, **pam_vas** compares usernames and passwords to previously cached values. User password hashes are cached in a secure file in **/var/state /vas/authcache/authcache.vdb**, which has permissions of 0600 (readable and writable only by the owner of the file) and is owned by root. Though SHA1 is a strong hash algorithm it is important to protect the **authcache** file in the same way as you should protect **/etc/shadow**.

On some systems it might be preferable to disable caching of usernames and password hashes. To do this, add **no_disconnected** to the **pam_vas** auth entry of the appropriate PAM configuration file.

NSS Configuration

The Name Service Switch (NSS) is a UNIX and Linux subsystem that allows the administrator to configure the data source that is used to obtain password entries, group entries, hosts, and other information. More information regarding the UNIX and Linux NSS subsystem can be found by consulting the system administration documentation for each of the operating systems.

nsswitch.conf

The configuration for the NSS subsystem is found in **/etc/nsswitch.conf**. The following is an example of the contents of a typical **nsswitch.conf** before configuring SCO Authentication:

```
passwd:     files nis
shadow:     files nis
group:      files nis
hosts:      files dns
ethers:     files
netmasks:   files
networks:   files
protocols:  files nis
rpc:        files
services:   files nis
```

In this example, note the state of the passwd and group.

The first string of each line is the database name followed by a colon. The strings that follow the database names are the names of the NSS modules that are used to resolve requests for information from that database. In the example above, passwd entries are first obtained using the "files" module that retrieves information from the **/etc/passwd** file. If the requested **passwd** entry is not found in **/etc/passwd**, the NIS module is called which attempts to find the entry using the NIS protocol.

Configuring NSS with vastool

NSS is configured using **vastool configure nss** or as a part of the **vastool join** process. By default, **vastool configure nss** inserts configuration for the "vas" module so that it is called directly after the files module. For example, running the following command as root:

```
# vastool configure nss
```

Modifies the "passwd" and "group" lines in **/etc/nsswitch.conf** so that they appear as follows:

```
passwd:     files vas nis
groups:     files vas nis
```

With this new configuration, the **vas** module is called to resolve all passwd and group entries that not are not found in the **/etc/passwd** and **/etc/group**.

The order in which the modules are listed is important. By default, the **vas** module is listed after the files module and before the nis module. This ordering means that local accounts take precedence over Active Directory accounts with the same name, UID, or GID.

Reverting NSS Configuration Changes

The SCO Authentication configuration for the NSS subsystem can be reverted using **vastool**. This removes the **nss_vas** module from the passwd and group lines in **/etc/nsswitch.conf**. The command is as follows (run as **root**):

```
# vastool unconfigure nss
```

Restarting Services

After changing the **nsswitch.conf** file on Solaris we recommend that certain services be restarted. For example, **dtlogin** on Solaris does not immediately acknowledge changes

made to **nsswitch.conf** and does not allow Active Directory users to log in until it is restarted. As a troubleshooting step, it might be necessary to restart other services on Solaris if Active Directory users can not log in after making changes to the NSS configuration.

4 UNIX and Linux Users and Groups

With SCO Authentication you can manage UNIX user accounts and UNIX groups and passwords from Active Directory using the Users and Computers snapin or the flexible command line utilities. Information in this section addresses the following concepts:

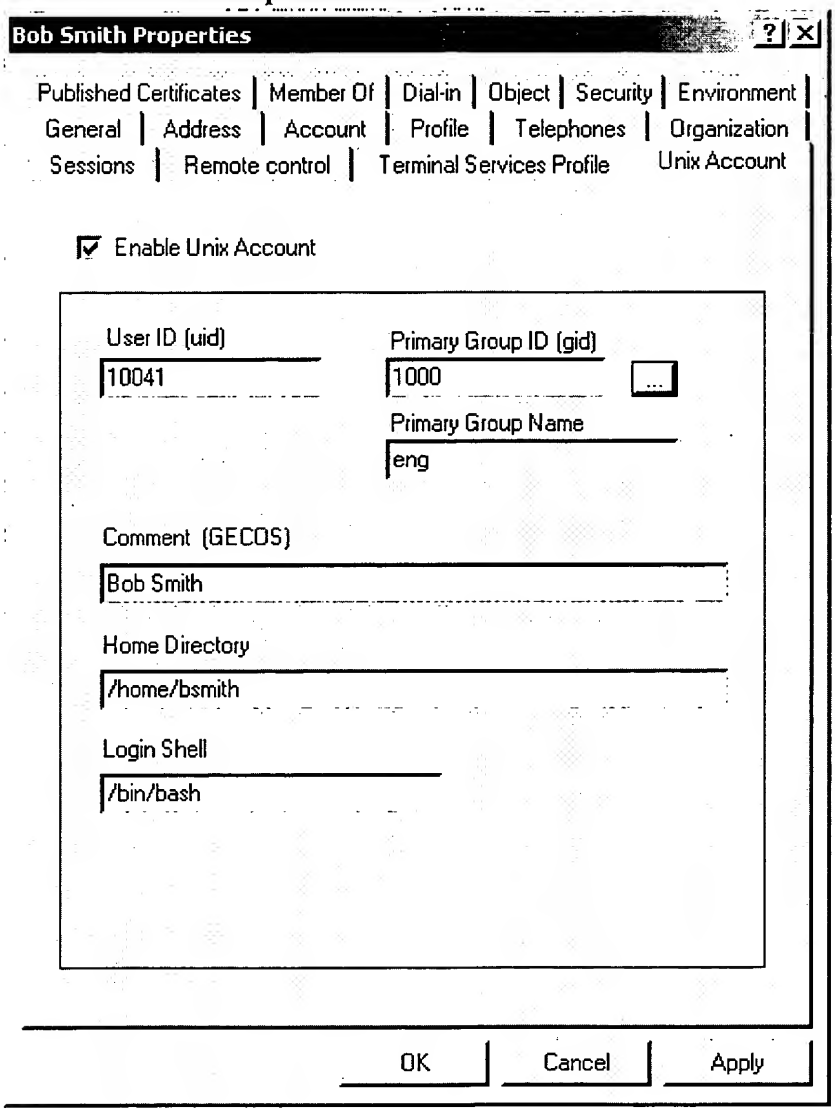
Managing UNIX User Accounts

Using the Users and Computers Snapin

After installing the Users and Computers Snapin extension, a **UNIX Account** tab

appears in the properties of all Active Directory users as shown in Figure 12 on page 50:

Figure 12. UNIX Account Properties



Note: If the UNIX Account tab does not appear in the User Properties dialog, review the installation steps outlined in the *SCO Authentication Installation and Configuration Guide* to ensure that the Users and Computers Snapin extension has been installed.

Enable UNIX Account - This checkbox is used to enable and disable UNIX and Linux accounts. Enabling a UNIX account for the first time generates default values for the UNIX account fields. Disabling a UNIX or Linux account causes the Login Shell to be set to **/bin/false** which prohibits the user from logging in to UNIX and Linux machines. For more information, see "Disabling UNIX Accounts" on page 54.

User ID - This field is used to set the numeric UNIX or Linux User ID or UID. It should be set to a numeric value between 1000 and the maximum UID for your platform. When the UNIX account is enabled for the first time, a default value is generated that is one greater than the highest UID previously used by users in the same Active Directory organizational unit. If it is the first UNIX account to be enabled in the organizational unit then the default value will be 1000.

Primary Group ID- This field is the UNIX or Linux User Group ID or user GID that is used when determining the group ownership of files that are created by the user. The **Browse** button allows the administrator to look through and select the Primary Group ID from the list of UNIX enabled groups. However, it is not required that the Primary Group ID be set to a value from this list.

Primary Group Name- If the Primary Group ID is set to the GID value of a UNIX enabled group account in Active Directory, then the Primary Group Name field displays the group name. If the Primary Group ID is set to a value found in the **/etc/group** (such as the value for the local "users" group) then Primary Group Name displays the text "Unknown Group" in the Properties dialog.

Comment (GECOS) - This is a free form field that is usually used to record the user's full name and other information (such as phone number and office location). The only restriction is that this field must not contain a colon (:) character. The default value is the user's full name.

Home Directory - This is the UNIX user's home directory. If the home directory does not exist when the user logs in to a machine for the first time, it can be created by the **pam_vas** module. However, since a great deal of UNIX user profile information (for example, desktop environment, shell profiles, and application specific settings) are saved in the user home directory it is possible to store home directories on a network

file system such as NFS. Storing home directories on a distributed file system is especially desirable in environments where UNIX and Linux users log in to multiple workstations. Automount utilities can then be configured so that the user's same home directory information is available in the same directory location across multiple UNIX and Linux machines.

Login Shell - This is the shell that is executed when the user logs in using a terminal-based login. The default value for Login Shell is **/bin/bash**. If you do not have this shell on the systems the user is logging into, you must change this setting to a valid login shell.

Since shells reside in different directories (such as **/usr/local/bin**) on different UNIX and Linux distributions we recommend that you create a symlink to allow commonly used shells to be found in the same location across multiple UNIX and Linux machines.

Managing User Accounts from the UNIX and Linux Command Line

Using the **vastool** command line utility you can create users, delete users, and list user information.

To create a user, use the **vastool create** command. For example, the following command would create the new user "wynn":

```
$ vastool create wynn
```

This creates a user in Active Directory that does not have its UNIX Account enabled. To create a user that has its UNIX account enabled by default, you need to pass in a string formatted like the entries in **/etc/passwd**. You would do this as follows:

```
$ vastool create -i "wynn:x:1003:1000:Wynn:/home/wynn:/bin/bash" wynn
```

It is important that you specify a unique UID. **vastool** assumes that the administrator is aware of the potential UID conflicts and does not request additional information.

To delete a user, use **vastool delete**. For example the following command would delete the "wynn" user:

```
$ vastool delete wynn
```

To list users, use **vastool list users**. For example, the following command would list all the users with UNIX accounts enabled:

```
$ vastool list users

    matt:VAS:1000:1000:Matt Peterson:/home/matt:/bin/bash
    wynn:VAS:1001:1000:Wynn Wilkes:/home/wynn:/bin/bash
```

The **vastool list** command without the **-l** option does not directly contact Active Directory but uses the **vascd** cache.

To bypass the cache and contact Active Directory directly, run the **vastool list** command with the **-l** option as follows:

```
$ vastool list -l users

    matt:x:1000:1000:Matt Peterson:/home/matt:/bin/bash
    wynn:x:1001:2000:Wynn Wilkes:/home/wynn:/bin/bash
```

It is significant to note that as a troubleshooting tool, it is possible to determine if the cache is out of date by comparing the differences between the output of **vastool list users** and **vastool list -l users**. If the cache is out of date, the output from those two commands will be different.

Using **vastool list users** with the **-l** option is not efficient when used repeatedly from scripts. You should not use the **-l** option in this case, but instead take advantage of the information stored in the **vascd** cache. For more information, see the "Troubleshooting" chapter of the *SCO Authentication Installation and Configuration Guide.*

Creating and Moving Users to Organizational Unit Containers

Use the Users and Computers snapin to create new users in any organizational unit (OU) and move existing users from one OU to another. For more information on how to use Organizational Units to keep track of GIDs, see, "UID and GID Management" on page 59.

An administrator can move a user object between organizational units within the same Active Directory domain without any problems. However, when a user is moved to a new domain, the Administrator should update the user's UPN to include the new domain. SCO Authentication uses the UPN to determine which domain the user is a member of, so it is important to keep this information consistent. Change the UPN using the Account tab on the user's properties page in the Active Directory Users and Computers Snapin. The domain the user is in should be updated to the user's new domain- this is done from the pull down menu that is part of the "User logon name:" section.

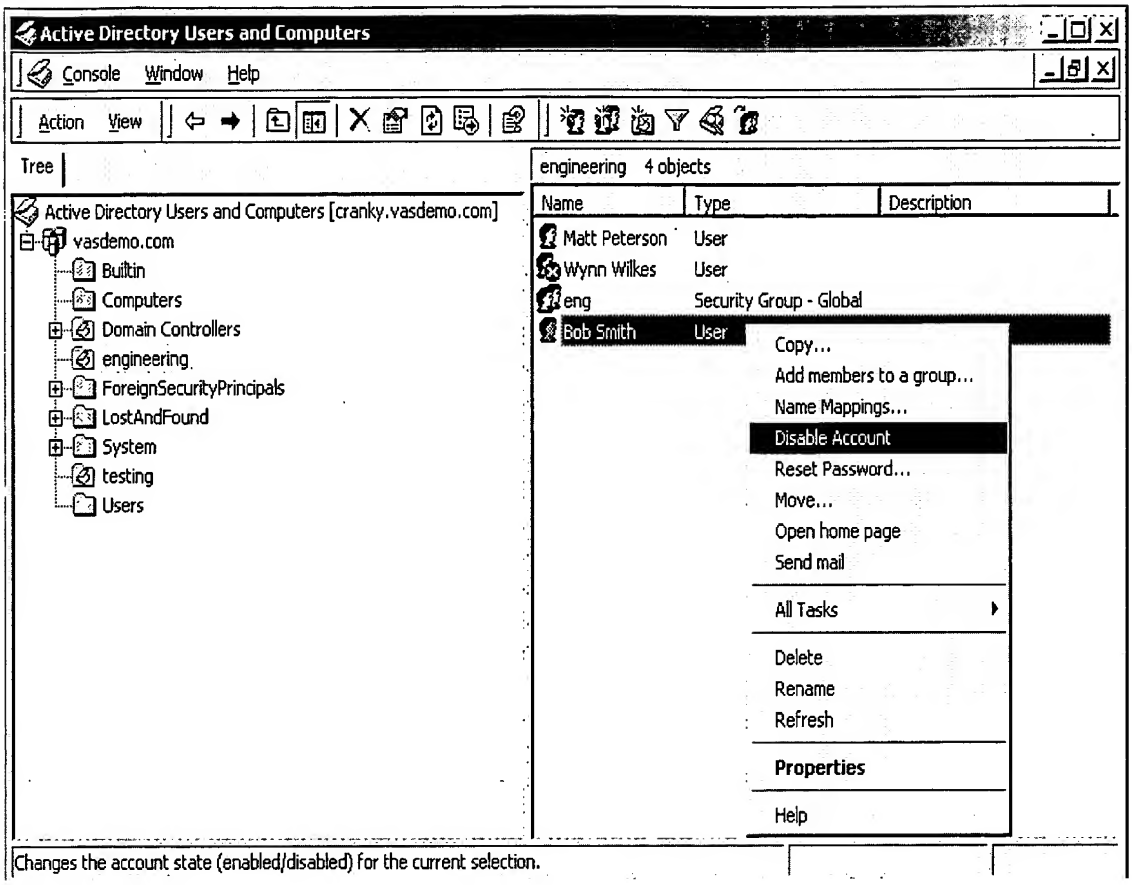
Disabling UNIX Accounts

When a user's UNIX account is disabled (by unchecking the **Enable UNIX Account** checkbox), the user's login shell is set to **/bin/false**. This prohibits the user from logging in via interactive login utilities. In technical terms, the check for whether or not a UNIX account is disabled is determined in a PAM "session" call. This means that non-interactive PAM enabled applications that do not use the PAM session interface would still be able to authenticate the user. An example of one such application is the POP3 daemon (**pop3d**) commonly distributed on Linux. Since **pop3d** obtains the username password via the POP3 protocol and is not concerned with giving the mail user a login shell there is no need to call the PAM session interface.

Therefore, in the case of **pop3d** (and other non-interactive services) a user would be allowed to authenticate even if their UNIX Account was disabled. To completely disable an account for interactive and non-interactive authentication, select the **Account is Disabled** setting from the **Account** tab or right click on the user and select **Disable**

Account in the Active Directory Users and Computers snapin.

Figure 13. Disabling a User Account



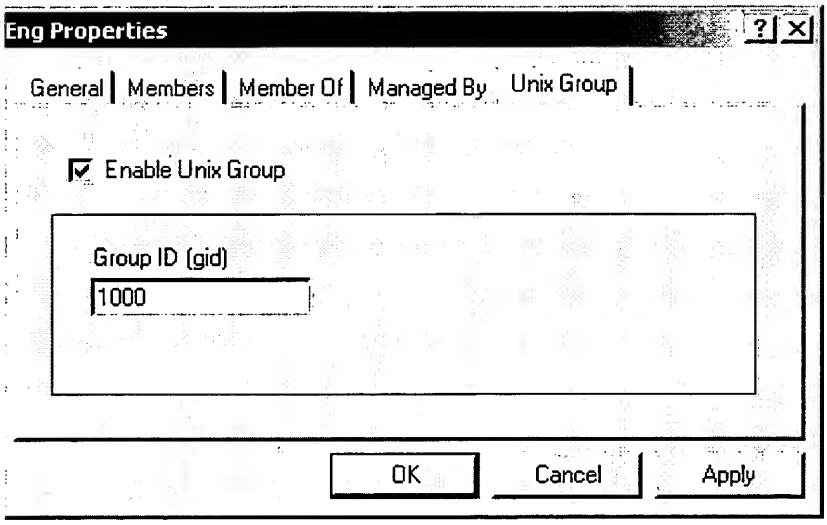
Another important property of a user with a disabled UNIX Account is that all of the UNIX account information remains valid even though the user is not able to log in on UNIX or Linux hosts. Retaining UNIX account information for disabled UNIX accounts is necessary in order for the ownership of files and directories to be properly listed when using **ls -l** command. If the user is removed entirely only then will the orphaned files list as being owned a numeric UID.

Managing UNIX Group Accounts

Using the Users and Computers Snapin

After installing the Users and Computers Snapin extension, a **UNIX Group** tab appears in the properties of all Active Directory groups as follows:

Figure 14. UNIX Group tab



Note: If the **Group** tab does not appear in the User Properties dialog, review the installation steps outlined in the *SCO Authentication Installation and Configuration Guide* and ensure that the Schema extension has been applied and that the Users and Computers Snapin extension has been installed.

The properties dialog includes the following:

Enable UNIX Group - This checkbox is used to enable and disable a UNIX Group. Enabling a UNIX group generate a default value for Group ID field. Disabling a UNIX group deletes the GID attribute which prohibits users who are members of the group from accessing UNIX files based on UNIX group permissions. For more information on disabled UNIX Groups, see, "Disabling UNIX Accounts" on page 54.

Group ID- This field is used to set the numeric Group ID or GID for the UNIX or Linux Group. It should be set to a numeric value between 1000 and the maximum group ID for your platform. When the UNIX Group is enabled for a UNIX or Linux group, a default value is generated that is one greater than the highest GID previously being used by UNIX Groups in the same Active Directory organizational unit. If it is the first UNIX or Linux Group to be enabled in the organizational unit then the default value will be 1000.

Managing Groups from the UNIX and Linux Command Line

Using the **vastool** command line utility you can create, delete, and list group information.

To create a group, use the **vastool create** command. For example, the following command creates the "eng" group:

```
$ vastool create -g eng
```

This creates the "eng" group in Active Directory but the group is not UNIX Group enabled. To create a group that is UNIX Group enabled by default, pass in a string formatted like the entries in **/etc/group**, as follows:

```
$ vastool create -i "eng:x:1003:" -g eng
```

It is important that you specify a unique GID. The use of **vastool** assumes that the administrator is aware of any potential GID conflicts and does not prompt for verification.

To delete a group, enter:

```
vastool delete -g eng
```

To list users use the **vastool list groups** command. For example, the following command would list all the groups with UNIX group enabled:

```
$ vastool list groups

    eng:VAS:2000:wynn,matt
    admin:VAS:2001:ben
```

This command does not directly contact Active Directory but uses the **vascd** cache.

To directly check Active Directory, run the **vastool list** command with the **-l** option, as follows:

```
$ vastool list -l groups

    eng:x:2000:wynn,matt
    admin:x:2001:erik
```

It is significant to note that as a troubleshooting tool, it is possible to determine if the cache is out of date by comparing the differences between the output of **vastool list groups** and **vastool list -l groups**. If the cache is out to date, those two commands will provide different output.

Using **vastool list groups** with the **-l** option is not efficient when used repeatedly from scripts. You should not use the **-l** option in this case, but instead take advantage of the information stored in the **vascd** cache.

Creating and Moving Groups to Organizational Unit Containers

Using the Users and Computers Snapin, new groups can be created in any Organizational Unit (OU) and existing groups can be moved from one OU to another. Since the distinguished name (DN) of the group is not relevant to the proper operation of SCO Authentication, you can create groups in or move groups to different OUs in order to organize groups in a way that makes administrative tasks intuitive.

UID and GID Management

When using SCO Authentication to manage users from Active Directory it is important to keep track of UNIX and Linux user UIDs and UNIX and Linux Group GIDs. The **UNIX Account** and **UNIX Group** tabs in the Users and Computers Snapin allows duplicate UIDs and GIDs to be assigned within a single Active Directory tree. Without careful planning, it is easy to lose track of which UIDs and GIDs have been used.

UID and GID Usage Organized by Container

To avoid UID and GID confusion it is recommended that users and groups be placed into Organizational Units (OUs) that are created according to an administrative model that makes sense to the network administrator. Since recommended default UIDs and GIDs are generated from the first objects in the same OU it is easy to assign a UID and GID range to an OU.

Avoiding Local UID and GID Duplication

When creating local user accounts in **/etc/passwd** and **/etc/group** it is important to be very careful not to duplicate UIDs or GIDs of Active Directory users and groups. The administrator needs to be especially careful when creating users using the **useradd** or **adduser** command. The reason is that unless the **-u** option is used to explicitly specify the UID, the default behavior of **useradd** uses NSS calls to determine the next highest UID. If the **nss_vas** module is configured in **nsswitch.conf** as it should be, the result will be a new local user in **/etc/passwd** with a UID that is one higher than highest UID in Active Directory. For security reasons, the **pam_vas** module does not allow duplications between UIDs in **/etc/passwd** and Active Directory. For security reasons, the **pam_vas** module does not allow Active Directory users to log in if they have a UID conflict with other users. Checks on UID duplication under 1000 is not performed.

Changing Passwords

One of the primary features of SCO Authentication is that the same user name and password can be used to login to Windows, UNIX, and Linux workstations. In conjunction with this feature it is also possible for a user to change their password from Windows, UNIX, and Linux workstations using the normal password change utilities.

As of this writing, the **passwd** command on SuSE, UnitedLinux powered distributions, and Solaris displays the following error when using the system passwd change utility:

```
passwd: Changing password for Supported configurations for
passwd management are as follows:

    passwd: files
    passwd: files ldap
    passwd: files nis
    passwd: files nisplus
    passwd: compat
    passwd: compat AND
    passwd_compat: ldap OR
    passwd_compat: nisplus
    Please check your /etc/nsswitch.conf file
    Permission denied
```

As a work around, SCO Authentication users should use **vastool passwd** to change their passwords that are stored Active Directory. Local users should use the **passwd -r files** command. Users on other platforms such as Red Hat should use the standard passwd change utility.

A user with Active Directory administrative privileges can change other users' passwords with the **vastool passwd** command, as follows:

```
$ vastool passwd user
```

Changing Passwords on Windows

Users may change their password using the standard Windows password change dialog. To open the password change dialog press **Ctrl-Alt-Del** and click on the **Change Password** button. After changing a password in Windows the new password is immediately applicable to UNIX and Linux logins.

Importing Users and Groups

For environments that have pre-existing **/etc/passwd** files and **/etc/group** files either locally or through NIS, **vastool** provides migration tools to import those users and groups into Active Directory using **vastool load**.

The **vastool load** command can either read a file that follows the **/etc/passwd** or **/etc/group** format, or it can read from "stdin" (piped output from another command).

Before importing a passwd file or a group file, make sure that the import will not cause a UID or GID conflicts with accounts that are already in Active Directory. The **vastool load** command does not check for those conflicts before creating the user and group objects.

Important: You can't use **vastool load** to import users or groups whose names are duplicates of those already in the Active Directory database.

Also, note that only users with the appropriate administrative rights can create users and groups. So if your current login session is not as an administrative user, use the **vastool -u** option to specify the administrative user name.

To import a users file without using "stdin", run:

```
$ vastool -u user load -f /tmp/new-vas-users.txt users
```

This creates all of these users in the default Users container in Active Directory. Use the **-c** option to specify a container in which to create users. The following is an example of importing users into a specific OU:

```
$ vastool -u user load -f /tmp/new-vas-user.txt \

-c "OU=eng,DC=example,DC=com" users
```

One limitation to importing users is the inability to import users' current passwords. Active Directory cannot accept the encrypted password hashes that are normally stored in passwd files. The **vastool load** command either generates a random password for each user which is then saved to a file that the administrator can use to notify new users and let them know what their new password is, or else the administrator can set a common default password for all newly imported user accounts with the "*-p password*" parameter.

After importing the users, you can then view these user accounts and their UNIX account information in the Active Directory Users and Computers console.

Note: Users are forced to change their passwords when they first login, by default.

To import groups, run:

```
$ vastool load -f /tmp/new-vas-groups.txt groups
```

As with the **vastool load users** command, these groups are created in the default users container. The same **-c** option applies for loading groups as with users, as follows:

```
$ vastool load -f /tmp/new-vas-groups.txt \
-c "OU=eng,DC=example,DC=com" groups
```

One important note about importing groups is that the group membership list is stored in the Active Directory Group Object's members attribute. The values stored here must be distinguished names (DNs). In order to create the group object with it's existing members, **vastool** must look up each user's DN, meaning that the user must already exist in Active Directory. So when importing users and groups, you must import the users first, then the groups, otherwise your group membership lists will not be imported correctly.

Workstation Access Control

In some environments there might be sensitive data on machines that are running SCO Authentication so the administrator might not want all SCO Authentication users to be able to log on to those machines. The **pam_vas** module allows administrators to have fine-grained control over which SCO Authentication users have access to a given system.

Note: Computer access control only works with Active Directory user accounts.

By default, each time a user is successfully authenticated, **pam_vas** checks to see if access should be allowed by using information recorded in **/etc/opt/vas/users.allow** and **/etc/opt/vas/users.deny**.

Valid entries in these files include user principal names (UPNs), group names, and realm names. UPNs are formatted as, **principal_name@domain** and realm names are formatted as, **@realm**. Any entry that does not have the '@' character is interpreted as a group name. Lines beginning with a hash (#) are ignored. For more information on UPNs, see "Multiple Domain Deployment" on page 77.

In all cases, **users.allow** takes precedence over the **users.deny**. For example, if a user is explicitly allowed, but explicitly denied by having the user's UPN in both files, then the user is granted access.

A user is granted access if one of the following is true:

- The **users.allow** file contains the users's UPN.
- The **users.allow** file contains a group the user is a member of and the **users.deny** file does not contain the user's UPN.
- The **users.allow** file contains the realm of the user, and the **users.deny** file does not contain either the user's UPN or a group the user belongs to.
- If the **users.allow** file is empty or does not exist, and the **users.deny** file does not deny the user in anyway- explicitly, by the group, or by the realm.
- If **users.allow** and **users.deny** do not exist.

In all other cases, the user is denied access. If either the **users.allow** or **users.deny** file is empty, they are treated the same as if they do not exist.

For example, if you had a sensitive server that you only wanted administrators to be able to log in to, you would create a **users.allow** that would look like the following:

```
## Only let these users log in.
matt@example.com
wynn@example.com
```

Where matt and wynn are users with administrator privileges.

If you wanted all users except for two to log in, do not have a **users.allow** file but rather create a **users.deny** file that would look like the following:

```
## These users are not allowed to log in.
don@example.com
rob@example.com
```

Where don and rob are users you want to lock out.

It is possible to disable these checks in **pam_vas** on a service by service basis. This might be necessary on sensitive servers that run non-login services that still need to authenticate all of your SCO Authentication users, but only allow certain users to have shell access on the server. To do this, use the "no_access_check" option in the PAM configuration file. For example, you could disable this check when using an IMAP server by modifying the following auth entry in the **/etc/pam.d/imapd** file:

```
/opt/vas/lib/security/pam_vas.so
```

to look like:

```
auth  [ignore=ignore success=done default=die] \
/opt/vas/lib/security/pam_vas.so no_access_check
```

5 Advanced SCO Authentication Configurations

On Linux distributions and the Solaris system, applications like Apache and Samba use the Pluggable Authentication Modules system (PAM) to integrate with SCO Authentication. In general, PAM includes a set of authentication modules that allow you to configure the type of authentication you want to use. You specify the authentication modules each application should use by identifying them in special configuration files within
/etc/pam.d.

Note: You must restart PAM-enabled applications once they are configured in order for them to interact with SCO Authentication.

Obtaining the PAM Module for Apache

Apache is PAM-enabled when it is run with the Apache PAM module. The PAM modules for Apache are available at:

http://pam.sourceforge.net/mod_auth_pam/

There are two versions available, one for Apache 1.3.x (mod_auth_pam-1.1.1) and another for use with Apache 2.0 (mod_auth_pam-2.0-1.1.1). Instructions for building the modules are included with each of the two source code tarballs.

In addition to a full C/C++ toolchain (including the compilers and libraries) and the Apache development files, **apxs** and **instdso.sh** must also be installed on the build system. On Linux, these three requirements are typically satisfied when the **apache-devel** or **httpd-devel** package is installed.

Configuring Apache to Run with SCO Authentication

The following instructions describe how to password-protect the documents that are in Apache's default web page directory using PAM and SCO Authentication.

The instructions require that you create and edit several files. Exact instructions on creating and editing files are dependent on the editor you use. Make sure you have an editor available that you are comfortable with before you start this procedure set.

Using UnitedLinux 1.0 and SuSE 8.1

The Apache 1.3 web server is included in the default installation of distributions powered by UnitedLinux.

On SuSE 8.1 and platforms powered by UnitedLinux 1.0, complete the following:

1. Make sure that **mod_auth_pam.so** is in **/usr/lib/apache**:

   ```
   cp mod_auth_pam.so /usr/lib/apache
   ```

   The module, **mod_auth_pam.so,** enables Apache to use PAM so it can run with SCO Authentication.

2. Change to the **/etc/httpd/modules** directory as follows:

   ```
   cd /etc/httpd/modules
   ```

3. Create a file named **mod_auth_pam** and include the following lines:

```
Variable:HTTPD_SEC_MOD_PAM
Text:PAM
LoadModule:LoadModule pam_auth_module \
/usr/lib/apache/mod_auth_pam.so
AddModule:AddModule mod_auth_pam.c
```

4. Open the file **/etc/sysconfig/apache** and add the following line to the end of the file:

```
HTTPD_SEC_MOD_PAM=yes
```

5. Open the file, **/etc/httpd/httpd.conf** and locate the following line:

```
<Directory "/srv/www/htdocs">
```

A few lines below, edit the following line:

```
AllowOverride None
```

to look like this:

```
AllowOverride AuthConfig
```

6. Change to the directory **/srv/www/htdocs**.

7. Create a file named **.htaccess** (make sure to include the leading period) and include the following lines:

```
AuthType Basic
AuthName "Secured"
require valid-user
```

Where `require valid-user` specifies that any valid user has access. Use this line to determine who has access to the directory **/srv/www/htdocs** and its recursive subdirectories. For example, `require matt` specifies that `matt` is the only user allowed access to **/srv/www/htdocs**. The entry, `require group acct` only allows access to members of the UNIX group named `acct`.

8. Create a backup copy of your existing **/etc/httpd/httpd.conf** file, then run the SuSEconfig utility against the **/etc/sysconfig/apache** file as follows:

```
cp /etc/httpd/httpd.conf /etc/httpd/httpd.conf-`date`
/sbin/SuSEconfig --force --module apache
```

9. Set up the Apache PAM configuration file to use SCO Authentication:

```
/opt/vas/bin/vastool configure pam httpd
```

10. Restart the Apache web server as follows:

```
/etc/init.d/apache restart
```

At this point, if you access the Apache server using a web browser, the prompt for a username and password appears. The user authentication information that you enter needs to match the user authentication information stored in Active Directory.

Using SuSE 8.0

Complete the following to configure Apache1.3 to run with SCO Authentication on SuSE 8.0:

1. Open **/etc/sysconfig/apache** and make sure the HTTPD_SEC_MOD_PAM line is set to `yes` as follows:

```
HTTPD_SEC_MOD_PAM=yes
```

2. Open the file, **/etc/httpd/httpd.conf** and locate the following line:

   ```
   <Directory "/srv/www/htdocs">
   ```

   A few lines below, edit the following line:

   ```
   AllowOverride None
   ```

   to look like this:

   ```
   AllowOverride AuthConfig
   ```

3. Change to the directory **/srv/www/htdocs**.

4. Create a file named **.htaccess** (make sure to include the leading period) and include the following lines:

   ```
   AuthType Basic
   AuthName "Secured"
   require valid-user
   ```

   Where `require valid-user` specifies that any valid user has access. Use this line to determine who has access to the directory **/srv/www/htdocs** and its recursive subdirectories. For example, `require matt` specifies that `matt` is the only user allowed access to **/srv/www/htdocs**. The entry, `require group acct` only allows access to members of the UNIX group named `acct`.

5. Create a backup copy of your existing **/etc/httpd/httpd.conf** file, then run the SuSEconfig utility against **httpd.conf** as follows:

   ```
   cp /etc/httpd/httpd.conf /etc/httpd/httpd.conf-`date`
   ```

   ```
   /sbin/SuSEconfig --force --module apache
   ```

6. Set up the Apache PAM configuration file to use SCO Authentication:

   ```
   /opt/vas/bin/vastool configure pam httpd
   ```

7. Restart the Apache web server as follows:

   ```
   /etc/init.d/apache restart
   ```

   At this point, if you access the Apache server using a web browser, the prompt for a username and password appears. The user authentication information that you enter needs to match the user authentication information stored in Active Directory.

Using Red Hat 7.3 (including Advanced Server)

Note: After following these instructions, you can no longer use Red Hat's Apache Configuration Tool. If you run that application, your SCO Authentication customizations to Apache will be wiped out. According to Red Hat, there is no way to avoid this dilemma.

On Red Hat 7.3 and Advanced Server, complete the following to configure Apache 1.3 to run with SCO Authentication:

1. Make sure that **mod_auth_pam.so** is in **/usr/lib/apache** as follows:

   ```
   cp mod_auth_pam.so /usr/lib/apache
   ```

   The module, **mod_auth_pam.so,** enables Apache to use PAM so it can run with SCO Authentication.

2. Open the file **/etc/httpd/conf/httpd.conf** and add the following line at the end of the list of `LoadModule` lines (the last `Loadmodule` line might be enclosed in `<IfDefine>...</IfDefine>` tags; if this is the case, add the following line *after* the last `</IfDefine>` closing tag) as follows:

   ```
   LoadModule pam_auth_module modules/mod_auth_pam.so
   ```

3. In the same **httpd.conf** file add the following line at the end of the `AddModules` lines (the last `AddModule` line might be enclosed in `<IfDefine>...</IfDefine>` tags; if this is the case, add the following line *after* the last `</IfDefine>` closing tag) as follows:

```
AddModule mod_auth_pam.c
```

4. In the same **httpd.conf** file locate the following line:

```
<Directory "/var/www/html">
```

A few lines below, edit the following line:

```
AllowOverride None
```

to look like this:

```
AllowOverride AuthConfig
```

5. Change to the directory **/var/www/html**.

6. Create a file named **.htaccess** (make sure to include the leading period) and include the following lines:

```
AuthType Basic
AuthName "Secured"
require valid-user
```

Where `require valid-user` specifies that any valid user has access. Use this line to determine who has access to the directory **/srv/www/htdocs** and its recursive subdirectories. For example, `require matt` specifies that `matt` is the only user allowed access to **/srv/www/htdocs**. The entry, `require group acct` only allows access to members of the UNIX group named `acct`.

7. Set up the Apache PAM configuration file to use SCO Authentication:

```
/opt/vas/bin/vastool configure pam httpd
```

8. Restart the Apache web server as follows:

   ```
   /etc/init.d/httpd restart
   ```

   At this point, if you access the Apache server using a web browser, the prompt for a username and password appears. The user authentication information that you enter needs to match the user authentication information stored in Active Directory.

Using Red Hat 8.0 and 9.0

Note: After following these instructions, you can no longer use Red Hat's Apache Configuration Tool. If you run that application, your SCO Authentication customizations to Apache will be wiped out. According to Red Hat, there is no way to avoid this dilemma.

The following instructions describe how to password-protect the documents in Apache's default web page directory using PAM and SCO Authentication. Red Hat 8.0 uses Apache 2.x, (see, "Obtaining the PAM Module for Apache" on page 65) so you need to build the Apache PAM module package that is used with that version of Apache.

On Red Hat 8.0, complete the following:

1. Make sure that **mod_auth_pam.so** is in **/usr/lib/httpd/modules** (or **/etc/httpd/modules** which is a symbolic link to the **/usr/lib/httpd/modules**) as follows:

   ```
   cp mod_auth_pam.so /usr/lib/httpd/modules
   ```

   and

   ```
   cp mod_sys_group.so /usr/lib/httpd/modules
   ```

2. Open the file **/etc/httpd/conf/httpd.conf** and add the following line at the end of the list of `LoadModule` lines (the last `Loadmodule` line might be enclosed in `<IfDefine>...</IfDefine>` tags; if this is the case, add the following line *after* the last `</IfDefine>` closing tag) as follows:

   ```
   LoadModule auth_pam_module modules/mod_auth_pam.so

   LoadModule auth_sys_group modules/mod_sys_group.so
   ```

3. In the same **httpd.conf** file locate the following line:

   ```
   <Directory "/var/www/html">
   ```

   A few lines below, edit the following line:

   ```
   AllowOverride None
   ```

   to look like this:

   ```
   AllowOverride AuthConfig
   ```

4. Change to the directory **/var/www/html**.

5. Create a file named **.htaccess** (make sure to include the leading period) and include the following lines:

   ```
   AuthType Basic
   AuthName "Secured"
   require valid-user
   ```

   Where `require valid-user` specifies that any valid user has access. Use this line to determine who has access to the directory **/var/www/html** and its recursive subdirectories. For example, `require matt` specifies that `matt` is the only user allowed access to **/var/www/html**. The entry, `require group acct` only allows access to members of the UNIX group named `acct`.

6. Set up the Apache PAM configuration file to use SCO Authentication:

   ```
   /opt/vas/bin/vastool configure pam httpd
   ```

7. Restart the Apache web server as follows:

   ```
   /etc/init.d/httpd restart
   ```

   At this point, if you access the Apache server using a web browser, the prompt for a username and password appears. The user authentication information that you enter needs to match the user authentication information stored in Active Directory.

Using SCO Authentication with Samba

Using SCO Authentication with Samba requires that you use **vastool timesync** to synchronize the computer running Samba with Active Directory server. For more information on synchronizing time using the **vastool** command line utility, see the **vastool** man page.

The following instructions apply to all supported Linux distributions.

To get Samba 2.2x and 3 running with SCO Authentication, complete the following:

1. Set up the PAM configuration to use SCO Authentication by entering:

   ```
   vastool configure pam samba
   ```

2. Join Samba to the NETBIOS domain used by the Windows server that's running with Active Directory by entering:

   ```
   smbpasswd -j NETBIOS_domain -r adserv -U matt
   ```

   Where:

   NETBIOS_domain is the workgroup name that Windows networking uses as your system name.

   adserv is the hostname or IP address of the Windows server in the sample network with Active Directory running.

matt is a user with Domain Admin privileges on the Active Directory server in the example.

3. Join the server running Samba to the SCO Authentication/Active Directory domain (this is not the NETBIOS domain of the Windows server), by entering:

   `vastool -u matt join -f example.com`

   Where:

   matt is a user with admin privileges.

   -f forces the creation of a new computer object when a computer object for the Samba machine already exists in Active Directory.

   example.com is the name of the Active Directory domain in the sample network.

4. Set up the **/etc/samba/smb.conf** file using `vastool` as follows:

   `vastool -u matt configure samba`

   Where *matt* is a user with Domain Admin privileges on the Active Directory server in the example.

5. Restart Samba. Use the instructions for your distributions.

   For UnitedLinux and SuSE, enter the following:

   ```
   /etc/init.d/smb restart

   /etc/init.d/nmb restart
   ```

   For Red Hat, enter the following:

   ```
   /etc/init.d/smb restart
   ```

Using SCO Authentication with SSH

You can use SCO Authentication with SSH (the OpenSSH version, not the proprietary SSH Communications Security version) without further configuration changes. However, if you want to force a password change at login, use the following instructions. Making this change creates a potential security risk when using unaudited PAM modules in your PAM stack for local user authentication.

Platforms supported as SCO Authentication clients that do not use PAM are not covered by these instructions.

1. Ensure that the following SSH server options are set in **sshd_config** (located in **/etc** or **/etc/ssh** depending on the platform being used):

   ```
   PAMAuthenticationViaKbdInt  yes

   UsePrivilegeSeparation  no
   ```

   Note: If these lines already exist in **sshd_config** but are commented out (the line starts with "#"), you must uncomment them (remove the "#" at the beginning of the lines) before saving **sshd_config**.

2. Re-start the SSH server as follows:

   ```
   /etc/init.d/sshd stop

   /etc/init.d/sshd start
   ```

With these steps completed, users who have accounts in Active Directory can log into the server via SSH using Active Directory authentication information.

6 Deployment Strategies

Multiple Domain Deployment

It is not uncommon to have an entire business operating within one Active Directory forest that contains only one Active Directory domain. As a business expands, the IT infrastructure requires that more domains be added to the forest to help manage the expansion. Obviously, the authentication system used by the company must be capable of expanding in parallel.

This section provides the following:

- Identifies the criteria for multiple domain deployments of SCO Authentication.
- Identifies the pros and cons of synchronizing SCO Authentication data to the Global Catalog.
- Explains how UNIX groups are handled across domains.

Criteria for Multiple Domain Deployment

SCO Authentication can provide authentication services over multiple domains as long as certain criteria are met. These criteria are addressed in the following:

- Users must use a User Principal Name (UPN) to log in and authenticate across domains. The UPN is formatted like an email address and can map to the users email address. The UPN format is **principal_name@domain**. For example,

  `matt@example.com`

  Where:

  matt is a principal_name in the sample network.

  example.com is the domain name in the sample network.

  Note: The UPN is only required for cross domain authentication. Users do not need to use a UPN when authenticating to a local domain.

- If a user requires membership in a particular UNIX group that is not resident on the local domain but is on another domain within the forest, create an identical user account for that user on the remote domain and then add the account to the target UNIX group. This is because a user's "local" account group memberships are not recognized in other domains. For more information, see "Working with Nested Groups" on page 78.

- Your DNS server must be set up to allow dynamic updates.

- The Active Directory SRV records must exist within DNS. If your DNS server is set up to allow dynamic updates, Active Directory should have added its own SRV records automatically.

Working with Nested Groups

SCO Authentication only supports "local" groups with no nesting. If you embed "global" or "universal" groups within local groups, only the top-level "local" group will be seen by SCO Authentication, regardless of whether or not the nested group originates in the "local" domain. Keep this in mind as you set up groups for UNIX users.

Using UID Name Spaces

When a user account in Active Directory is UNIX enabled (For information on UNIX enabling an account, see "Managing UNIX User Accounts" on page 49), a suggested value for the user's UID appears in the **User ID (uid)** field on the UNIX Account Properties dialog. This information is provided by the Users and Computers snapin. The suggestion is based on the highest previously-assigned UID in the same organizational unit, incremented by one.

As long as you are using one Active Directory server in your Active Directory forest and as long as you always accept the suggested UID values there should be no risk of ID conflicts. However, when you are using more than one Active Directory server in your forest you need to be a bit more careful about conflicting UIDs. If suggested UID values are accepted on two or more Active Directory servers in a single forest, it is highly likely that new UID assignments will conflict almost from the beginning.

In the world of UNIX, system administrators who face these types of scenarios deal with the problem by splitting up their allocations of UIDs into smaller ranges or name spaces, each of which is assigned to a specific organizational unit of the network. For example, an administrator might assign UID name space 1000-2000 to the accounting organizational unit, 2001-3000 to the engineering organizational unit, and 3001-4000 to the sales organizational unit.

To achieve similar name space functionality for user accounts in Active Directory, set the UID for the first user in the organizational unit to equal the first, lowest available value of that name space. For example, using the UID name space example listed earlier, the administrator would set up the first user account from the sales organizational unit with UID 3001, the first user account from the engineering organizational unit with UID 2001, and the first user account from the accounting organizational unit with UID 1001.

From that point forward, each new user in each organizational unit will receive a suggested UID that is one higher than the UID for the previously entered user account. Continuing with the examples listed earlier, the second sales user account would get UID 3002, the second engineering user account would get UID 2002, and the second accounting user account would get UID 1002.

There is no hard, fast rule for splitting up UIDs into name spaces. There are 65,535 UIDs available on a typical UNIX machine (This number might be different on the platform you are using. Be sure to check your system documentation.) so name spaces can be split into much larger chunks if necessary.

Note that the UID values that the Users and Computers snapin offers are just suggestions. There is nothing stopping an administrator from assigning a different number to a new user account. However, accepting the suggested UID values is recommended.

Migration from UNIX Kerberos to SCO Authentication

SCO Authentication provides the same Active Directory Kerberos inter-operability as UNIX Kerberos without the complexity that goes along with it. Setting up Kerberos inter-operation with Active Directory is as simple as installing SCO Authentication on a UNIX computer and joining the computer to the Active Directory realm.

SCO Authentication in Place of Straight Kerberos

Because SCO Authentication utilizes Kerberos v.5 and because its configuration and **keytab** files (**/etc/opt/vas/vas.conf** and **/etc/opt/vas/host.keytab**, respectively) mirror the structure and functionality of those used by Kerberos v.5, migrating Kerberos functionality to SCO Authentication is very simple.

Assuming SCO Authentication is already installed, complete the following:

1. If necessary, move the existing **/etc/krb5.conf** file out of the way.
2. Create a symbolic link pointing from **/etc/krb5.conf** to **/etc/opt/vas/vas.conf** as follows:

```
ln -s /etc/opt/vas/vas.conf /etc/krb5.conf
```

3. Store the SCO Authentication realm information in **vas.conf**.

Because SCO Authentication utilizes DNS SRV records by default, your **/etc/opt/vas/vas.conf** file does not contain any realm configurations after installing SCO Authentication. When using SCO Authentication for straight Kerberos inter-operation you must store the realm configuration SCO Authentication knows about in the **vas.conf**.

To do so, run the following command:

```
/opt/vas/bin/vastool realms cache tovas
```

Note: If Active Directory servers are created or removed from your network and the DNS SRV records change, you might need run this command again.

With that done, UNIX-enabled user accounts in Active Directory will be able to authenticate to the UNIX machine through Kerberos v.5. This applies not only to shell logins but also to Kerberos v.5 enabled applications.

Group Migration

If there are groups on the UNIX system that you would like to migrate to Active Directory, SCO Authentication provides a simple UNIX group migration procedure:

1. Create a file containing all of the group information that needs to be migrated.

2. Load the groups from the group file into Active Directory.

The group file is a file that closely resembles **/etc/group** on the UNIX system. The format of each colon-delimited line follows that of **/etc/group**:

```
Group:x:GID:Member1,Member2,
```

The second field - which might contain a group password - is ignored. The last field can optionally contain a comma-delimited list of users who are members of the group. If you choose to load a group that utilizes this last field, the user accounts that you specify must already exist in Active Directory before loading the group. For details on loading

users from UNIX into Active Directory, see "User Migration" on page 83.

An edited copy of the UNIX machine's **/etc/group** file provides a suitable group file as long as the following conditions are met:

- Do not try to load any named groups that already exist in Active Directory. Active Directory group names are not case-sensitive, so trying to load a group named **cvsusrs** from UNIX into Active Directory fails if Active Directory already has a group named **CVSusrs**, **CVSUSRS**, **cvsusrs**, or **CvsUsrs**.
- Do not try to load service groups like "ftp" or "mail". Leave them on the UNIX machine.
- Do not duplicate GID values in the group file that are already used with other UNIX groups in Active Directory. Adjust the GID values as necessary in either Active Directory or the group file so there are no conflicts.

When you are ready to load the UNIX groups from the group file into Active Directory, use **vastool load**:

```
vastool -u admin -f group_file groups
```

Where:

admin is the name of a user with Domain Admin privileges in Active Directory.

group_file is the name of the file containing the UNIX group data you want to load.

Optionally, you can specify a specific container on the **vastool load** command line using the **-c** parameter:

```
vastool -u admin -f group_file -c container groups
```

Where container is the distinguished name (DN) of the container. For example, OU=dev,DC=example,DC=com.

User Migration

If there are user accounts on the UNIX system that you would like to migrate to Active Directory, SCO Authentication provides a simple UNIX user migration procedure:

1. Create a file containing all of the account information that needs to be migrated.

2. Load the users from the user file into Active Directory.

The user file is a file that closely resembles /etc/passwd on the UNIX system. The format of each colon-delimited line follows that of /etc/passwd:

```
Login:x:UID:GID:Comment:Home_directory:Login_shell
```

The second field - which usually contains a password - is ignored. For explanations of what these fields should contain see, "Using the Users and Computers Snapin" on page 56.

An edited copy of the UNIX machine's **/etc/passwd** file provides a suitable user file as

long as the following conditions are met:

- Do not try to load any accounts that already exist in Active Directory. Active Directory user names are not case-sensitive, so trying to load a user named **wynn** from UNIX into Active Directory fails if Active Directory already has an account named **Wynn**, **WYNN**, **wynn**, or **WynN**.
- Do not try to load service users like "ftp" or "mail". Leave them on the UNIX machine.
- Do not duplicate UID or GID values in the user file that are already used with other UNIX users and groups in Active Directory. Adjust the UID and GID values as necessary in either Active Directory or the user file so there are no conflicts.

As previously mentioned, the second field of each line in the user file - the one that contains a password in **/etc/passwd** - is ignored. This means that UNIX passwords does not migrate to Active Directory. Because of this, you must assign a default password when loading users into Active Directory using SCO Authentication.

When you are ready to load the UNIX users from the user file into Active Directory, use **vastool load**:

```
vastool -u admin -f user_file -p password users
```

Where:

admin is the name of a user with Domain Admin privileges in Active Directory.

user_file is the name of the file containing the UNIX user data you want to load.

password is the default password assigned to all of the migrated UNIX user accounts.

Optionally, you can force the migrated UNIX users to change their passwords the next time they log in by passing the **-x** parameter on the **vastool load** command line, and you can specify a specific container on the using the **-c** parameter:

```
vastool -u admin -f user_file -c container -x users
```

Where *container* is the distinguished name (DN) of the container. For example, OU=dev,DC=example,DC=com.

7 NIS Migration

The Network Information System (NIS) is a mechanism that is used to distribute information throughout a network such as user accounts, network addresses, and configuration files. Processes normally access this information using internal interfaces in the operating systems's system libraries, or else they use the `ypcat` program to print out the NIS information.

NIS information is stored in a NIS Map. A NIS map is data that represents a configuration file like **/etc/hosts**. SCO Authentication allows for these NIS maps to be stored in Active Directory so that they then can be distributed using the **vascd** daemon and the **vastool ypcat** command.

Because the NIS Maps must be imported into Active Directory, the **vastool nis-import** command creates NIS Map objects in Active Directory. For example, to create a hosts map, first edit a file to contain the information formatted in the same way that **/etc/hosts** is formatted. Then run the command:

```
$ vastool nis-import -f /tmp/eng-hosts.txt hosts
```

The new NIS Map is created in the default Computers organizational unit (OU). This is important since the **vascd** running on a given computer only searches the OU that the computer is in for NIS Maps. This is valuable in that it gives you a way to emulate NIS domains, which allow you to have different name spaces for the NIS Maps.

The **vastool nis-import** command also accepts a **-c** *container* option to store the NIS Map in any Active Directory Container. This allows you to arrange different groups' computers.This makes it possible to distribute different maps of the same name to different containers of computers.

To access the NIS map contents, you can use the **vastool ypcat** command, or the **/opt/vas/bin/ypcat** script. The **ypcat** script is simply a wrapper script that calls the **vastool ypcat** command. This allows the administrator to symlink the SCO Authentication **ypcat** script into the standard location for **ypcat** to allow existing applications to use the new **ypcat** without modifications.

The **vastool ypcat** command works by sending an IPC request to **vascd**, which then uses Kerberos/LDAP to download the map contents. Then **vastool** parses the map contents and prints the map information to the local console (stdout).

It is important to note that this NIS Map support does not use the NIS network protocol in any way. The NIS Map information is still transmitted using the secure Kerberos and LDAP communication that **vascd** uses to get all of its information. Using **vascd** also allows NIS Map information to be cached.

An example of accessing the hosts map out of Active Directory is:

```
$ vastool ypcat -k hosts
```

The **vastool ypcat** command supports the same **-k** and **-x** options as **ypcat**. These can be used to look at the NIS Map keys and view the Map nickname translation table.

Currently SCO Authentication supports the following maps:

passwd

group

hosts

ethers

aliases

auto.master

auto.home

services

protocols

rpc

netgroup

You can not import passwd or group maps. This should be done using the **vastool load** command instead. Any map starting with "auto." will be parsed as an "autofs" map.

In order to view maps such as **hosts.byname**, run:

```
$ vastool ypcat -k hosts.byname
```

A ftp Man Pages

Note: For the most update version of the man pages, check the product CD.

ftp(1) Man Page

ftp (1)

NAME

ftp

ARPANET file transfer program

SYNOPSIS

`ftp [-t] [-v] [-d] [-i] [-n] [-g] [-p] [-l] {host}ftp`

Description

`ftp` is the user interface to the ARPANET standard File Transfer Protocol. The program allows a user to transfer files to and from a remote network site.

Modifications has been made so that it almost follows the ftpsec Internet draft.

Options may be specified at the command line, or to the command interpreter.

`-t`	Enables packet tracing.
`-v`	Verbose option forces ftp to show all responses from the remote server, as well as report on data transfer statistics.
`-n`	Restrains ftp from attempting "auto-login" upon initial connection. If auto-login is enabled, ftp will check the `.netrc` (see

below) file in the user's home directory for an entry describing an account on the remote machine. If no entry exists, ftp will prompt for the remote machine login name (default is the user identity on the local machine), and, if necessary, prompt for a password and an account with which to login.

`-i` Turns off interactive prompting during multiple file transfers.

`-p` Turn on passive mode.

`-d` Enables debugging.

`-g` Disables file name globbing.

`-l` Disables command line editing.

The client host with which ftp is to communicate may be specified on the command line. If this is done, ftp will immediately attempt to establish a connection to an FTP server on that host; otherwise, ftp will enter its command interpreter and await instructions from the user. When ftp is awaiting commands from the user the prompt 'ftp>' is provided to the user. The following commands are recognized by ftp:

`! [command [args]]`

Invoke an interactive shell on the local machine. If there are arguments, the first is taken to be a command to execute directly, with the rest of the arguments as its arguments.

`$ macro-name [args]`

Execute the macro macro-name that was defined with the macdef command. Arguments are passed to the macro unglobbed.

`account [passwd]`

Supply a supplemental password required by a remote system for access to resources once a login has been successfully com-

pleted. If no argument is included, the user will be prompted for an account password in a non-echoing input mode.

`append local-file [remote-file]`

Append a local file to a file on the remote machine. If remote-file is left unspecified, the local file name is used in naming the remote file after being altered by any ntrans or nmap setting. File transfer uses the current settings for type, format, mode, and structure.

`ascii` Set the file transfer type to network ASCII. This is the default type.

`bell` Arrange that a bell be sounded after each file transfer command is completed.

`binary` Set the file transfer type to support binary image transfer.

`bye` Terminate the FTP session with the remote server and exit ftp. An end of file will also terminate the session and exit.

`case` Toggle remote computer file name case mapping during mget commands. When case is on (default is off), remote computer file names with all letters in upper case are written in the local directory with the letters mapped to lower case.

`cd remote-directory`

Change the working directory on the remote machine to remote-directory.

`cdup` Change the remote machine working directory to the parent of the current remote machine working directory.

`chmod mode file-name`

Change the permission modes of the file *file-name* on the remote sytem to *mode*.

`close` Terminate the FTP session with the remote server, and return to the command interpreter. Any defined macros are erased.

`cr` Toggle carriage return stripping during ascii type file retrieval. Records are denoted by a carriage return/linefeed sequence during ascii type file transfer. When cr is on (the default), carriage returns are stripped from this sequence to conform with the UNIX single linefeed record delimiter. Records on nonâUNIX remote systems may contain single linefeeds; when an ascii type transfer is made, these linefeeds may be distinguished from a record delimiter only when cr is off.

`delete remote-file`

Delete the file *remote-file* on the remote machine.

`debug [debug-value]`

Toggle debugging mode. If an optional *debug-value* is specified it is used to set the debugging level. When debugging is on, ftp prints each command sent to the remote machine, preceded by the string '-->'

`dir [remote-directory] [local-file]`

Print a listing of the directory contents in the directory, *remote-directory* , and, optionally, placing the output in *local-file*. If interactive prompting is on, ftp will prompt the user to verify that the last argument is indeed the target local file for receiving dir output. If no directory is specified, the current working directory on the remote machine is used. If no local file is specified, or *local-file* is -, output comes to the terminal.

`disconnect` A synonym for *close*.

`form format` Set the file transfer form to format. The default format is "file".

`get remote-file [local-file]`

Retrieve the *remote-file* and store it on the local machine. If the local file name is not specified, it is given the same name it has on the remote machine, subject to alteration by the current case, ntrans, and nmap settings. The current settings for type, form, mode, and structure are used while transferring the file.

`glob`

Toggle filename expansion for mdelete, mget and mput. If globbing is turned off with glob, the file name arguments are taken literally and not expanded. Globbing for mput is done as in csh(1). For mdelete and mget, each remote file name is expanded separately on the remote machine and the lists are not merged. Expansion of a directory name is likely to be different from expansion of the name of an ordinary file: the exact result depends on the foreign operating system and ftp server, and can be previewed by doing 'mls remote-files -'. As a security measure, remotely globbed files that starts with '/' or contains '../', will not be automatically received. If you have interactive prompting turned off, these filenames will be ignored. Note: mget and mput are not meant to transfer entire directory subtrees of files. That can be done by transferring a tar(1) archive of the subtree (in binary mode).

`hash`

Toggle hash-sign ("#") printing for each data block transferred. The size of a data block is 1024 bytes.

`help [command]`

Print an informative message about the meaning of *command*. If no argument is given, ftp prints a list of the known commands.

`idle [seconds]`

Set the inactivity timer on the remote server to seconds seconds. If *seconds* is omitted, the current inactivity timer is printed.

`lcd [directory]`

Change the working directory on the local machine. If no *directory* is specified, the user's home directory is used.

`ls [remote-directory] [local-file]`

Print a listing of the contents of a directory on the remote machine. The listing includes any system-dependent information that the server chooses to include; for example, most UNIX systems will produce output from the command 'ls -l'. (See also nlist.) If *remote-directory* is left unspecified, the current working directory is used. If interactive prompting is on, ftp will prompt the user to verify that the last argument is indeed the target local file for receiving ls output. If no local file is specified, or if *local-file* is '-', the output is sent to the terminal.

`macdef macronname`

Define a macro. Subsequent lines are stored as the macro *macro-name*; a null line (consecutive newline characters in a file or carriage returns from the terminal) terminates macro input mode. There is a limit of 16 macros and 4096 total characters in all defined macros. Macros remain defined until a close command is executed. The macro processor interprets '$' and '\' as special characters. A '$' followed by a number (or numbers) is replaced by the corresponding argument on the macro invocation command line. A '$' followed by an 'i' signals that macro processor that the executing macro is to be looped. On the first pass '$i' is replaced by the first argument on the macro invocation command line, on the second pass it is replaced by the second argument, and so on. A '\' followed by any character is replaced by that character. Use the '\' to prevent special treatment of the '$'.

`mdelete [remote-files]`

Delete the remote-files on the remote machine.

`mdir remote-files local-file`

Like dir, except multiple remote files may be specified. If interactive prompting is on, ftp will prompt the user to verify that the last argument is indeed the target local file for receiving mdir output.

`mget remote-files`

Expand the *remote-files* on the remote machine and do a get for each file name thus produced. See glob for details on the filename expansion. Resulting file names will then be processed according to case, ntrans, and nmap settings. Files are transferred into the local working directory, which can be changed with 'lcd directory'; new local directories can be created with '! mkdir directory'.

`mkdir directory-name`

Make a directory on the remote machine.

`mls remote-files local-file`

Like nlist, except multiple remote files may be specified, and the *local-file* must be specified. If interactive prompting is on, ftp will prompt the user to verify that the last argument is indeed the target local file for receiving mls output.

`mode [mode-name]`

Set the file transfer mode to *mode-name*. The default mode is "stream" mode.

`modtime file-name`

Show the last modification time of the file on the remote machine.

`mput local-files`

Expand wild cards in the list of local files given as arguments and do a put for each file in the resulting list. See glob for details of filename expansion. Resulting file names will then be processed according to ntrans and nmap settings.

`newer file-name`

Get the file only if the modification time of the remote file is more recent that the file on the current system. If the file does not exist on the current system, the remote file is considered newer. Otherwise, this command is identical to *get*.

`nlist [remote-directory] [local-file]`

Print a list of the files in a directory on the remote machine. If *remote-directory* is left unspecified, the current working directory is used. If interactive prompting is on, ftp will prompt the user to verify that the last argument is indeed the target local file for receiving nlist output. If no local file is specified, or if *local-file* is -, the output is sent to the terminal.

`nmap [inpattern outpattern]`

Set or unset the filename mapping mechanism. If no arguments are specified, the filename mapping mechanism is unset. If arguments are specified, remote filenames are mapped during mput commands and put commands issued without a specified remote target filename. If arguments are specified, local filenames are mapped during mget commands and get commands issued without a specified local target filename. This command is useful when connecting to a non-UNIX remote computer with different file naming conventions or practices. The mapping follows the pattern set by *inpattern* and *outpattern*. [*Inpattern*] is a template for incoming filenames (which may have already been processed according to the ntrans and case settings). Variable templating is accomplished by including the sequences '$1', '$2', ..., '$9' in

inpattern. Use '\' to prevent this special treatment of the '$' character. All other characters are treated literally, and are used to determine the nmap [*inpattern*] variable values. For example, given inpattern $1.$2 and the remote file name "mydata.data", $1 would have the value "mydata", and $2 would have the value "data". The *outpattern* determines the resulting mapped filename. The sequences '$1', '$2', ..., '$9' are replaced by any value resulting from the *inpattern* template. The sequence '$0' is replace by the original filename. Additionally, the sequence '[*seq1*, *seq2*]' is replaced by [*seq1*] if *seq1* is not a null string; otherwise it is replaced by *seq2*. For example, the command

nmap $1.$2.$3 [$1,$2].[$2,file]

would yield the output filename "myfile.data" for input filenames "myfile.data" and "myfile.data.old", "myfile.file" for the input filename "myfile", and "myfile.myfile" for the input filename ".myfile". Spaces may be included in *outpattern*, as in the example: 'nmap $1 sed "s/ *$//" > $1'. Use the '\' character to prevent special treatment of the '$','[','[', and ',' characters.

`ntrans [inchars [outchars]]`

Set or unset the filename character translation mechanism. If no arguments are specified, the filename character translation mechanism is unset. If arguments are specified, characters in remote filenames are translated during mput commands and put commands issued without a specified remote target filename. If arguments are specified, characters in local filenames are translated during mget commands and get commands issued without a specified local target filename. This command is useful when connecting to a non-UNIX remote computer with different file naming conventions or practices. Characters in a filename matching a character in *inchars* are replaced with the corresponding character in *outchars*. If the character's position in *inchars* is longer than the length of *outchars*, the character is deleted from the file name.

`open host [port]`

Establish a connection to the specified *host* FTP server. An optional port number may be supplied, in which case, ftp will attempt to contact an FTP server at that port. If the auto-login option is on (default), ftp will also attempt to automatically log the user in to the FTP server (see below).

`passive`

Toggle passive mode. If passive mode is turned on (default is off), the ftp client will send a PASV command for all data connections instead of the usual PORT command. The PASV command requests that the remote server open a port for the data connection and return the address of that port. The remote server listens on that port and the client connects to it. When using the more traditional PORT command, the client listens on a port and sends that address to the remote server, who connects back to it. Passive mode is useful when using ftp through a gateway router or host that controls the directionality of traffic. (Note that though ftp servers are required to support the PASV command by RFC 1123, some do not.)

`prompt`

Toggle interactive prompting. Interactive prompting occurs during multiple file transfers to allow the user to selectively retrieve or store files. If prompting is turned off (default is on), any mget or mput will transfer all files, and any mdelete will delete all files.

`proxy ftp-command`

Execute an ftp command on a secondary control connection. This command allows simultaneous connection to two remote ftp servers for transferring files between the two servers. The first proxy command should be an open, to establish the secondary control connection. Enter the command "proxy ?" to see other ftp commands executable on the secondary connection. The following commands behave differently when prefaced by proxy: open will not define new macros during the auto-login process, close will not erase existing macro definitions, get and mget transfer

files from the host on the primary control connection to the host on the secondary control connection, and put, mput, and append transfer files from the host on the secondary control connection to the host on the primary control connection. Third party file transfers depend upon support of the ftp protocol PASV command by the server on the secondary control connection.

`put local-file [remote-file]`

Store a local file on the remote machine. If *remote-file* is left unspecified, the local file name is used after processing according to any ntrans or nmap settings in naming the remote file. File transfer uses the current settings for type, format, mode, and structure.

`pwd` Print the name of the current working directory on the remote machine.

`quit` A synonym for bye.

`quote arg1 arg2 ...`

The arguments specified are sent, verbatim, to the remote FTP server.

`recv remote-file [local-file]`

A synonym for get.

`reget remote-file [local-file]`

Reget acts like get, except that if *local-file* exists and is smaller than *remote-file*, *local-file* is presumed to be a partially transferred copy of *remote-file* and the transfer is continued from the apparent point of failure. This command is useful when transferring very large files over networks that are prone to dropping connections.

`remotehelp [command-name]`

Request help from the remote FTP server. If a *command-name* is specified it is supplied to the server as well.

`remotestatus [file-name]`

With no arguments, show status of remote machine. If *file-name* is specified, show status of *file-name* on remote machine.

`rename [from] [to]`

Rename the file *from* on the remote machine, to the file *to*.

`reset` Clear reply queue. This command re-synchronizes command/reply sequencing with the remote ftp server. Resynchronization may be necessary following a violation of the ftp protocol by the remote server.

`restart marker`

Restart the immediately following get or put at the indicated *marker*. On UNIX systems, marker is usually a byte offset into the file.

`rmdir directory-name`

Delete a directory on the remote machine.

`runique` Toggle storing of files on the local system with unique filenames. If a file already exists with a name equal to the target local filename for a get or mget command, a ".1" is appended to the name. If the resulting name matches another existing file, a ".2" is appended to the original name. If this process continues up to ".99", an error message is printed, and the transfer does not take place. The generated unique filename will be reported. Note that runique will not affect local files generated from a shell command (see below). The default value is off.

`send local-file [remote-file]`

A synonym for put.

`sendport` Toggle the use of PORT commands. By default, ftp will attempt to use a PORT command when establishing a connection for each data transfer. The use of PORT commands can prevent delays when performing multiple file transfers. If the PORT command fails, ftp will use the default data port. When the use of PORT commands is disabled, no attempt will be made to use PORT commands for each data transfer. This is useful for certain FTP implementations which do ignore PORT commands but, incorrectly, indicate they've been accepted.

`site arg1 arg2 ...`

The arguments specified are sent, verbatim, to the remote FTP server as a SITE command.

`size file-name`

Return size of *file-name* on remote machine.

`status` Show the current status of ftp.

`struct [struct-name]`

Set the file transfer *structure* to *struct-name*. By default "stream" structure is used.

`sunique` Toggle storing of files on remote machine under unique file names. Remote ftp server must support ftp protocol STOU command for successful completion. The remote server will report unique name. Default value is off.

`system` Show the type of operating system running on the remote machine.

`tenex` Set the file transfer type to that needed to talk to TENEX machines.

`trace` Toggle packet tracing.

`type [type-name]`

Set the file transfer type to *type-name*. If no type is specified, the current type is printed. The default type is network ASCII.

`umask [newmask]`

Set the default umask on the remote server to *newmask*. If *newmask* is omitted, the current umask is printed.

`user user-name [password] [account]`

Identify yourself to the remote FTP server. If the *password* is not specified and the server requires it, ftp will prompt the user for it (after disabling local echo). If an *account* field is not specified, and the FTP server requires it, the user will be prompted for it. If an *account* field is specified, an account command will be relayed to the remote server after the login sequence is completed if the remote server did not require it for logging in. Unless ftp is invoked with "auto-login" disabled, this process is done automatically on initial connection to the FTP server.

`verbose` Toggle verbose mode. In verbose mode, all responses from the FTP server are displayed to the user. In addition, if verbose is on, when a file transfer completes, statistics regarding the efficiency of the transfer are reported. By default, verbose is on.

`? [command]` A synonym for help.

The following commands can be used with ftpsec-aware servers.

`prot clear | safe | confidential | private`

Set the data protection level to the requested level.

The following command can be used with ftp servers that has implemented the KAUTH site command.

`kauth [principal]`

Obtain remote tickets.

Command arguments which have embedded spaces may be quoted with quote "" marks.

ABORTING A FILE TRANSFER

To abort a file transfer, use the terminal interrupt key (usually Ctrl-C). Sending transfers will be immediately halted. Receiving transfers will be halted by sending a ftp protocol ABOR command to the remote server, and discarding any further data received. The speed at which this is accomplished depends upon the remote server's support for ABOR processing. If the remote server does not support the ABOR command, an 'ftp>' prompt will not appear until the remote server has completed sending the requested file.

The terminal interrupt key sequence will be ignored when ftp has completed any local processing and is awaiting a reply from the remote server. A long delay in this mode may result from the ABOR processing described above, or from unexpected behavior by the remote server, including violations of the ftp protocol. If the delay results from unexpected remote server behavior, the local ftp program must be killed by hand.

FILE NAMING CONVENTIONS

Files specified as arguments to ftp commands are processed according to the following rules.

1. If the file name '-' is specified, the stdin (for reading) or stdout (for writing) is used.

2. If the first character of the file name is '|', the remainder of the argument is interpreted as a shell command. Ftp then forks a shell, using popen(3) with the argument supplied, and reads (writes) from the stdout (stdin). If the shell command includes spaces, the argument must be quoted; e.g. "" ls -lt"". A particularly useful example of this mechanism is: "dir more".

3. Failing the above checks, if "globbing" is enabled, local file names are expanded according to the rules used in the csh(1); c.f. the glob command. If the ftp command expects a single local file (.e.g. put), only the first filename generated by the "globbing" operation is used.

4. For mget commands and get commands with unspecified local file names, the local filename is the remote filename, which may be altered by a case, ntrans, or nmap setting. The resulting filename may then be altered if runique is on.

5. For mput commands and put commands with unspecified remote file names, the remote filename is the local filename, which may be altered by a ntrans or nmap setting. The resulting filename may then be altered by the remote server if sunique is on.

FILE TRANSFER PARAMETERS

The FTP specification specifies many parameters which may affect a file transfer. The type may be one of "ascii", "image" (binary), "ebcdic", and "local byte size" (for PDP-10's and PDP-20's mostly). Ftp supports the ascii and image types of file transfer, plus local byte size 8 for tenex mode transfers.

Ftp supports only the default values for the remaining file transfer parameters: mode, form, and struct.

THE .netrc FILE

The *.netrc* file contains login and initialization information used by the auto-login process. It resides in the user's home directory. The following tokens are recognized; they may be separated by spaces, tabs, or new-lines:

`machine name` Identify a remote machine *name*. The auto-login process searches the *.netrc* file for a machine token that matches the remote machine specified on the ftp command line or as an open command argument. Once a match is made, the subsequent *.netrc* tokens are processed, stopping when the end of file is reached or another machine or a default token is encountered.

`default` This is the same as machine *name* except that default matches any name. There can be only one default token, and it must be after all machine tokens. This is normally used as:

default login anonymous password user@site

thereby giving the user *automatic* anonymous ftp login to machines not specified in *.netrc*. This can be overridden by using the -n flag to disable auto-login.

`login name` Identify a user on the remote machine. If this token is present, the auto-login process will initiate a login using the specified *name*.

`password string`

Supply a password. If this token is present, the auto-login process will supply the specified string if the remote server requires a password as part of the login process. Note that if this token is present in the *.netrc* file for any user other than *anonymous*, ftp will abort the auto-login process if the *.netrc* is readable by anyone besides the user.

`account string`

Supply an additional account password. If this token is present, the auto-login process will supply the specified string if the remote server requires an additional account password, or the auto login process will initiate an ACCT command if it does not.

`macdef name` Define a macro. This token functions like the ftp macdef command functions. A macro is defined with the specified name; its contents begin with the next *.netrc* line and continue until a null line (consecutive new-line characters) is encountered.

ENVIRONMENT

Ftp utilizes the following environment variables.

`HOME` For default location of a *.netrc* file, if one exists.

SHELL For default shell.

See Also

ftpd(8)

RFC 2228

History

The ftp command appeared in 4.2BSD.

Bugs

Correct execution of many commands depends upon proper behavior by the remote server.

An error in the treatment of carriage returns in the 4.2BSD ascii-mode transfer code has been corrected. This correction may result in incorrect transfers of binary files to and from 4.2BSD servers using the ascii type. Avoid this problem by using the binary image type.

ftpd(8) Man Page

ftpd (8)

NAME

ftpd

Internet File Transfer Protocol server

SYNOPSIS

ftpd [-a *authmode] [-dilvU] [-g umask]{port}* [-T *maxtime-*

out] [-t timeout] [-u default umask] [-B | --bulitin-ls] [--good-chars=string]`ftpd`

Description

`Ftpd` is the Internet File Transfer Protocol server process. The server uses the TCP protocol and listens at the port specified in the "ftp" service specification; see services(5).

Available options:

`-a` Select the level of authentication required. Kerberised login can not be turned off. The default is to only allow kerberised login. Other possibilities can be turned on by giving a string of comma separated flags as argument to -a. Recognized flags are:

`plain`	Allow logging in with plaintext password. The password can be a(n) OTP or an ordinary password.
`otp`	Same as plain, but only OTP is allowed.
`ftp`	Allow anonymous login.
	The following combination modes exists for backwards compatibility:
`none`	Same as plain, ftp.
`safe`	Same as ftp.
`user`	Ignored

`-d` Debugging information is written to the syslog using LOG_FTP.

`-g` Anonymous users will get a umask of umask.

`-i` Open a socket and wait for a connection. This is mainly used for debugging when ftpd isn't started by inetd.

`-l` Each successful and failed ftp(1) session is logged using syslog

with a facility of LOG_FTP. If this option is specified twice, the retrieve (get), store (put), append, delete, make directory, remove directory and rename operations and their filename arguments are also logged.

`-p` Use port (a service name or number) instead of the default ftp/tcp.

`-T` A client may also request a different timeout period; the maximum period allowed may be set to timeout seconds with the -T option. The default limit is 2 hours.

`-t` The inactivity timeout period is set to timeout seconds (the default is 15 minutes).

`-u` Set the initial umask to something else than the default 027.

`-U` In previous versions of ftpd, when a passive mode client requested a data connection to the server, the server would use data ports in the range 1024..4999. Now, by default, if the system supports the IP_PORTRANGE socket option, the server will use data ports in the range 49152..65535. Specifying this option will revert to the old behavior.

`-v` Verbose mode.

`-B, --builtin-ls` use built-in ls to list files

`--good chars=string` allowed anonymous upload filename chars

The file `/etc/opt/vas/nologin` can be used to disable ftp access. If the file exists, ftpd displays it and exits. If the file `/etc/opt/vas/ftpwelcome` exists, ftpd prints it before issuing the "ready" message. If the file `/etc/opt/vas/motd` exists, ftpd prints it after a successful login.

The ftp server currently supports the following ftp requests. The case of the requests is ignored.

`Request Description`

ABOR	abort previous command
ACCT	specify account (ignored)
ALLO	allocate storage (vacuously)
APPE	append to a file
CDUP	change to parent of current working directory
CWD	change working directory
DELE	delete a file
HELP	give help information
LIST	give list files in a directory ("ls -lgA")
MKD	make a directory
MDTM	show last modification time of file
MODE	specify data transfer *mode*
NLST	give name list of files in directory
NOOP	do nothing
PASS	specify password
PASV	prepare for server-to-server transfer
PORT	specify data connection port
PWD	print the current working directory
QUIT	terminate session
REST	restart incomplete transfer

RETR	retrieve a file
RMD	remove a directory
RNFR	specify rename-from file name
RNTO	specify rename-to file name
SITE	non-standard commands (see next section)
SIZE	return size of file
STAT	return status of server
STOR	store a file
STOU	store a file with a unique name
STRU	specify data transfer *structure*
SYST	show operating system type of server system
STRU	specify data transfer *type*
USER	specify user name
XCUP	change to parent of current working directory (deprecated)
XCWD	change working directory (deprecated)
XMKD	make a directory (deprecated)
XPWD	print the current working directory (deprecated)
XRMD	remove a directory (deprecated)

The following commands are specified by RFC2228.

AUTH	authentication/security mechanism
ADAT	authentication/security data
PROT	data channel protection level
PBSZ	protection buffer size
MIC	integrity protected command
CONF	confidentiality protected command
CCC	clear command channel

The following non-standard or UNIX specific commands are supported by the SITE request.

UMASK	change umask, (e.g. SITE UMASK 002)
IDLE	set idle-timer, (e.g. SITE IDLE 60)
CHMOD	change mode of a file (e.g. SITE CHMOD 755 filename)
FIND	quickly find a specific file with GNU locate(1).
HELP	give help information.

The following Kerberos related site commands are understood.

KAUTH	obtain remote tickets.
KLIST	show remote tickets

The remaining ftp requests specified in Internet RFC 959 are recognized, but not implemented. MDTM and SIZE are not specified in RFC 959, but will appear in the next updated FTP RFC.

The ftp server will abort an active file transfer only when the

ABOR command is preceded by a Telnet "Interrupt Process" (IP) signal and a Telnet "Synch" signal in the command Telnet stream, as described in Internet RFC 959. If a STAT command is received during a data transfer, preceded by a Telnet IP and Synch, transfer status will be returned.

Ftpd interprets file names according to the "globbing" conventions used by csh(1). This allows users to utilize the metacharacters "*?[]{}~".

Ftpd authenticates users according to these rules.

1. If Kerberos authentication is used, the user must pass valid tickets and the principal must be allowed to login as the remote user.

2. The login name must be in the password data base, and not have a null password (if kerberos is used the password field is not checked). In this case a password must be provided by the client before any file operations may be performed. If the user has an OTP key, the response from a successful USER command will include an OTP challenge. The client may choose to respond with a PASS command giving either a standard password or an OTP one-time password. The server will automatically determine which type of password it has been given and attempt to authenticate accordingly. See otp(1) for more information on OTP authentication.

3. The login name must not appear in the file `/etc/opt/vas/ftpusers`.

4. The user must have a standard shell

returned by getusershell(3).

5. If the user name appears in the file `/etc/opt/vas/ftpchroot` the session's root will be changed to the user's login directory by chroot(2) as for an "anonymous" or "ftp" account (see next item). However, the user must still supply a password. This feature is intended as a compromise between a fully anonymous account and a fully privileged account. The account should also be set up as for an anonymous account.

6. If the user name is "anonymous" or "ftp", an anonymous ftp account must be present in the password file (user "ftp"). In this case the user is allowed to log in by specifying any password (by convention an email address for the user should be used as the password).

In the last case, ftpd takes special measures to restrict the client's access privileges. The server performs a chroot(2) to the home directory of the "ftp" user. In order that system security is not breached, it is recommended that the "ftp" subtree be constructed with care, consider following these guidelines for anonymous ftp.

In general all files should be owned by "root", and have non-write permissions (644 or 755 depending on the kind of file). No files should be owned or writable by "ftp" (possibly with exception for the `~ftp/incoming` , as specified below).

`~ftp` The "ftp" homedirectory should be owned by root.

`~ftp/bin` The directory for external programs (such

	as ls(1)). These programs must either be statically linked, or you must setup an environment for dynamic linking when running chrooted. These programs will be used if present:
`ls`	Used when listing files.
`compress`	When retrieving a filename that ends in *.Z*, and that file isn't present, ftpd will try to find the filename without *.Z* and compress it on the fly.
`gzip`	Same as compress, just with files ending in *.gz*.
`gtar`	Enables retrieval of whole directories as files ending in *.tar*. Can also be combined with compression. You must use GNU Tar (or some other that supports the -z and -Z flags).
`locate`	Will enable "fast find" with the SITE FIND command. You must also create a *locatedb* file in `~ftp/etc`.
`~ftp/etc`	If you put copies of the passwd(5) and group(5) files here, ls will be able to produce owner names rather than numbers. Remember to remove any passwords from these files. The file *motd*, if present, will be printed after a successful login.
`~ftp/dev`	Put a copy of /dev/null(7) here.
`~ftp/pub`	Traditional place to put whatever you want to make public.

If you want guests to be able to upload files, create a `~ftp/incoming` directory owned by "root", and group "ftp" with mode 730 (make sure "ftp" is member of group "ftp"). The following restrictions apply to anonymous users:

1. Directories created will have mode 700.

2. Uploaded files will be created with an umask of 777, if not changed with the -g option.

3. These command are not accessible: DELE, RMD, RNTO, RNFR, SITE UMASK, and SITE CHMOD.

4. Filenames must start with an alpha-numeric character, and consist of alpha-numeric characters or any of the following: + (plus), - (minus), = (equal), _ (underscore), . (period), and , (comma).

FILES

`/etc/opt/vas/ftpusers` Access list for users.

`/etc/opt/vas/ftpchroot` List of normal users who should be chroot'd.

`/etc/ftpwelcome` Welcome notice.

`/etc/motd` Welcome notice after login.

`/etc/nologin` Displayed and access refused.

`~/.klogin` Login access for Kerberos.

SEE ALSO

`ftp(1) otp(1) getusershell(3) ftpusers(5) syslogd(8)`

STANDARDS

RFC 959- FTP PROTOCOL SPECIFICATION

RFC 1938- OTP SPECIFICATION

RFC 2228- FTP Security Extensions

BUGS

The server must run as the super-user to create sockets with privileged port numbers. It maintains an effective user id of the logged in user, reverting to the super-user only when binding addresses to sockets. The possible security holes have been extensively scrutinized, but are possibly incomplete.

HISTORY

The ftpd command appeared in 4.2BSD.

ftpusers(5) Man Page

ftpusers (5)

NAME

ftpusers

FTP access list file

SYNOPSIS

```
ftpusers ftp users
```

DESCRIPTION

`/etc/ftpusers` contains a list of users that should be allowed or denied FTP access. Each line contains a user, optionally followed by 'allow' (anything but 'allow' is ignored). The semi-user '*' matches any user. Users that have an explicit 'allow', or that does not match any line, are allowed access. Anyone else is denied access.

Note that this is compatible with the old format, where this file contained a list of users

that should be denied access.

EXAMPLES

This will deny anyone but 'foo' and 'bar' to use FTP:

foo allow bar allow *

SEE ALSO

`ftpd(8)`

B login(1) Man Page

Note: For the most update version of the man pages, check the product CD.

login (1)

NAME

login

authenticate a user and start new session

SYNOPSIS

`login -fp {string} {hostname} {version} --help username login`

DESCRIPTION

The `login` program logs users into the system. It is intended to be run by system daemons like getty (8) or telnetd (8). If you are already logged in, but want to change to another user, you should use su (1).

A username can be given on the command line, else one will be prompted for.

A password is required to login, unless the`f` is given (indicating that the calling program has already done proper authentication). With `f` the user will be logged in without further questions.

For password authentication Kerberos 5, Kerberos 4 (if compiled in), OTP (if compiled in) and local passwords are supported. OTP will be used if the user is registered to use it, and `login` is given the option `-a` otp. When using OTP, a challenge is shown to the user.

Further options are:

`-a` string	Which authentication mode to use, the only supported value is currently "otp".
`-f`	Indicates that the user is already authenticated. This happens, for instance, when login is started by telnetd, and the user has proved authentic via Kerberos.
`-h` hostname	Indicates which host the user is logging in from. This is passed from telnetd, and is entered into the login database.
`-p`	This tells `login` to preserve all environment variables. If not given, only the TERM and TZ variables are preserved. It could be a security risk to pass random variables to `login` or the user shell, so the calling daemon should make sure it only passes "safe" variables.

The process of logging user in proceeds as follows.

First a check is made that logins are allowed at all. This usually means checking `/etc/nologin` . If it exists, and the user trying to login is not root, the contents is printed, and then login exits.

Then various system parameters are set up, like changing the owner of the tty to the user, setting up signals, setting the group list, and user and group id. Also various machine specific tasks are performed.

Next `login` changes to the users home directory, or if that fails, to `/` . The environment is setup, by adding some required variables (such as PATH), and also authentication related ones (such as KRB5CCNAME). If an environment file exists (`/etc/environment`), variables are set according to it.

If one or more login message files are configured, their contents is printed to the terminal.

If a login.time command is configured, it is executed. A logout time command can also

be configured, which makes `login` fork, and wait for the user shell to exit, and then run the command. This can be used to clean up user credentials.

Finally, the user's shell is executed. If the user logging in is root, and root's login shell does not exist, a default shell (usually `/bin/sh`) is also tried before giving up.

ENVIRONMENT

These environment variables are set by login (not including ones set by `/etc/environment`):

`PATH`	the default system path
`HOME`	the user's home directory (or possibly `/`)
`USER, LOGNAME`	both set to the username
`SHELL`	the user's shell
`TERM, TZ`	set to whatever is passed to login
`KRB5CCNAME`	if the password is verified via Kerberos 5, this will point to the credentials cache file
`KRBTKFILE`	if the password is verified via Kerberos 4, this will point to the ticket file

FILES

/etc/environment Contains a set of environment variables that should be set in addition to the ones above. It should contain sh-style assignments like "VARIABLE=value" . Note that they are not parsed the way a shell would. No variable expansion is performed, and all strings are literal, and quotation marks should not be used. Everything after a hash mark is considered a comment. The following are all different (the last will set the variable BAR, not FOO).

FOO="this is a string"

BAR= FOO='this is a string'

/etc/login.access login.access (5), /etc/login.conf This is a termcap style configuration file, that contains various settings used by `login` . Currently only the "default" capability record is used. The possible capability strings include:

environment

This is a comma separated list of environment files that are read in the order specified. If this is missing the default `/etc/environmenti` is used.

login_program

This program will be executed just before the user's shell is started. It will be called without arguments.

logout_program

This program will be executed just after the user's shell has terminated. It will be called without arguments. This program will be the parent process of the spawned shell.

motd

A comma separated list of text files that will be printed to the user's terminal before starting the shell. The string welcome works similarly, but points to a single file.

/etc/nologin.conf If it exists, login is denied to all but root. The contents of this file is printed before login exits. Other `login` programs typically print all sorts of information by default, such as last time you logged in, if you have mail, and system message files. This version of `login` does not, so there is no reason for .hushlogin files or similar. We feel that these tasks are best left to the user's shell, but the login_program facility allows for a shell independent solution, if that is desired.

EXAMPLES

A login.conf file could look like:

```
default:\
        :motd=/etc/motd,/etc/
        motd.local:
```

NOTES

This version of login contains added functionality for use with SCO Authentication. By default the `/login` will create users' home directories, prevent Active Directory users from logging in if they have a UID conflict, do access control based on, and allow disconnected authentication. These behaviors can all be configurable through which supports the following options in the [login] section. Note that the [login] section does not exist by default.

`realm_prompt` This option can be set to 0 or 1, 0 being the default. When set to 1, the password prompt for users will contain their whole user principal name and look like: 'Password for john@hisrealm.com: '. Note that enabling this can be a security hole since it can reveal valid account names. To enable this, add the following to:

`create_homedir`

This option can be set to 0 or 1, 1 being the default. When set to 1, a user's home directory will be created when they login. Setting this to 0 will disable that behavior and home directories will not be created.

`no_disconnected`

This option can be set to 0 or 1, 0 being the default. When set to 1, disconnected authentication will be disabled, and user passwords will not be cached.

`no_uidconflict_check`

This option can be set to 0 or 1, 0 being the default. When set to 1, the UID conflict check will not be performed when a user logs in, so if they do have a UID conflict with someone else, then they will still be able to login. Note that disabling the UID conflict check can create a security hole since users that share UID's will have access to each other's files.

`no_access_check`

This option can be set to 0 or 1, 0 being the default. When set to 1, login will not use the users.allow and users.deny files in to determine if users have shell access to the machine login is running on. For more information on how to use the users.allow and users.deny files, please see the pam_vas man page.

An example of what the [login] section should look like is:

[login] realm_prompt = 1 create_homedir = 0 no_disconnected = 1
no_uidconflict_check = 1 no_access_check = 1

Note that each of these settings can also be modified using the `vastool configure vas`.

SEE ALSO

su (1), login.access (5) getty (8) telnetd (8) vastool (1) pam_vas (5)

AUTHORS

This login program was written for the Heimdal Kerberos 5 implementation. The `login.access` code was written by Wietse Venema.

C pam_vas(5) Man Page

Note: For the most update version of the man pages, check the product CD.

pam_vas (5)

NAME

pam_vas

A PAM module for authenticating Active Directory users on Linux/UNIX computers via the Kerberos protocol.

SYNOPSIS

```
pam_vas <control-flags> /pam_vas.so [debug] [get_tgt]
[service=servicename] [helper_timeout=seconds]
[create_homedir] [no_access_check] [no_uidconflict_check]
[no_disconnected] [realm_prompt] <control-flags> /
pam_vas.so [debug] <control-flags> /pam_vas.so [debug]
[helper_timeout=seconds] <control-flags> /pam_vas.so
[debug]pam_vas
```

Description

pam_vas is the PAM component of the SCO Authentication client. When used in conjunction with the `vascd` daemon and the NSS module `nss_vas`, pam_vas enables Unix services to authenticate users whose credentials are stored in Active Directory-which acts as the Kerberos Key Distribution Center (KDC).

PAM enabled services are configured with four types of entries, auth, acct, password, and session. pam_vas can be used with all of these. The following is a list of each type of PAM configuration entry and what pam_vas can do for that type of PAM call:

auth — pam_vas will handle the authentication of a user name and password via the Kerberos protocol against Active Directory. pam_vas can also create a user's home directory if it does not

	exist, check for system UID conflicts, set up a user's Kerberos ticket cache, and check for computer access restrictions for the given user.
acct	pam_vas will check for user account restrictions set in Active Directory.
password	pam_vas will change a user's Active Directory password.
session	pam_vas will initialize a user's login session.

`vastool configure pam` will configure the system's PAM configuration. Do not be change the defaults created by `vastool` unless you really know what you are doing.

Control Flags

pam_vas has some special features which allow you to take advantage of the extended syntax for control flags in newer versions of PAM that are found in recent Linux distributions (this extended syntax is not supported in current versions of Solaris). pam_vas will ignore any calls for users that are not SCO Authentication users, i.e. system accounts. This allows you to use the following syntax:

[ignore=ignore success=done default=die]

"ignore=ignore" allows the PAM library to call other PAM modules in the stack if the username is not recognized by pam_vas as an Active Directory user. "success=done" instructs the PAM library to complete the authentication process if pam_vas is successful. "default=die" instruct the PAM library to error out if pam_vas returns an error condition. This control flag syntax is the default configuration on systems that support the /etc/pam.d/* PAM configuration. On platforms that use `/etc/pam.conf`, the "sufficient" control flag is used by default.

If the PAM library on your operating system (for example- Solaris 8 and 9), does not support this extended syntax, you will need to use the "sufficient" control flag. Unfortunately, this control flag will cause the PAM library to continue down the PAM stack when pam_vas fails which may result in multiple prompts to the user.

For example, if the PAM library does not support the extended control flag syntax, it is not possible to distinguish the difference between a local user login, and an Active Directory login with an incorrect password. The result is that if an Active Directory user mis-types their password, the PAM library will continue down the PAM stack and additionally allow the other PAM modules (such as pam_unix) to prompt the user for a password.

Note that PAM configuration file syntax can be quite complex, so make sure you know what you are doing before changing the default control flags that are configured by `vastool`.

Auth Arguments

These are the arguments that you can append on to the auth entries for pam_vas. By default, pam_vas auth entries are configured with the `get_tgt` and `create_homedir` arguments.

`debug` — Log debug messages to syslog. These will normally go to the secure system logs.

`get_tgt` — Obtains a Kerberos Ticket Granting Ticket (TGT) using the user's supplied credentials, and then uses that TGT to obtain a service ticket for the computer object in Active Directory. A Kerberos TGT is a ticket that is obtained with the user's password, and can then be used to obtain other tickets without having to reuse the user's password. In other words, the use of `get_tgt` saves the necessary TGT credentials for subsequent use of to kerberized "sign on" utilities.

Removing `get_tgt` option will cause pam_vas to obtain a service ticket directly using the user's password, without obtaining the TGT. The ticket is not stored anywhere on the filesystem, and is destroyed when the PAM session is ended. This results in less network traffic and load on the KDC. It is therefore recommended that you remove the `get_tgt` option when using pam_vas with a service that does not establish interactive login sessions, such as a web server or an IMAP server.

`service={service principal name}`

By default pam_vas will try to obtain a service ticket for your computer principal name. The computer principal name will be generated from the computer's hostname, or from the `host_principal_override` option in the `[libdefaults]` section. In the default case, the computer object for the local machine is the Service Principal used by pam_vas. The `service` allows the administrator to change the service name to be something other than the "host" principal. The `service` is an experimental option. Do not use it unless you know what you are doing.

`helper_timeout=`*`seconds`*

pam_vas uses an external program called `pam_vas_helper` to perform the Kerberos password change. There is a default timeout of 5 seconds, after which the external program will return. If using pam_vas in an environment where the network connection to the Active Directory KDC is very slow or when Active Directory contains a large number of users (+20000), then you can extend the timeout.

If the helper_timeout is too short then Active Directory users will be authenticated in disconnected mode against their cached password, which is stored as an SHA-1 hash.

`create_homedir`

Create the user's home directory if it does not exist. This will also copy the `/etc/skel` contents into the new home directory and setup the appropriate permissions.

`no_access_check`

Disables the workstation access control checks pam_vas performs. See the Computer Access Control section portion of his man page for more details.

`no_uidconflict_check`

Disables the UID conflict checking. UID checking will not allow any Active Directory user to login to the system if they have a UID conflict with another user. This does not affect local accounts. UID checking is performed for security reasons to prevent users from accidentally gaining access to files and resources they should not be able to.

UID checking is `not` performed for UID's under 1000. This allows administrators to setup accounts in Active Directory that could be used for root access.

`no_disconnected`

Disables disconnected authentication. Setting this option will also disable the caching of the user passwords hashes.

`realm_prompt` Adds the user's realm name to the password prompt. Note that that this is a potential security hole since it shows that a valid account name has been entered.

Acct Arguments

These are the arguments that can be append on to the acct entries for pam_vas. By default, pam_vas acct entries are not configured with any arguments.

`debug` Log debug messages to syslog. These will normally go to the secure system logs.

Password Arguments

These are the arguments that can be append on to the password entries for pam_vas. By default, pam_vas password entries are not configured with any arguments.

`debug` Log debug messages to syslog. These will normally go to the secure system logs.

`helper_timeout=`*`seconds`*

pam_vas uses an external program called `pam_vas_helper` to perform the Kerberos password change. There is a default timeout of 5 seconds, after which the external program will return. If using pam_vas in an environment where the network connection to the Active Directory KDC is very slow or when Active Directory contains a large number of users (+5000), then you can extend the timeout.

If the helper_timeout is too short then Active Directory users will authenticated in disconnected mode against previously recorded password hashes.

Session Arguments

These are the arguments that can be append on to the session entries for pam_vas. By default, pam_vas session entries are not configured with any arguments.

`debug` Log debug messages to syslog. These will normally go to the secure system logs.

Computer Access Control

It is possible to have fine grained control over which Active Directory users can login using pam_vas. This is especially useful when pam_vas is used on sensitive servers where shell access should only be granted to a few users. Note that computer access control functionality does not work with local user accounts, only with Active Directory user accounts.

By default, each time a user is successfully authenticated, pam_vas will check to see if access should be allowed by using information recorded in users.allow and user.deny.

Lines starting with '#' are comments. Valid entries are user principal names, groups, and realm names. User principal names will have the form of user@realm and realm names will have the form of @realm. Any entry that does not have the '@' character will be interpreted as a group name. A user will be granted access according to the following rules:

1. If the users.allow file contains the users's UPN.

2. If the users.allow file contains a group the user is a member of and the users.deny file does not contain the user's UPN.

3. If the users.allow file contains the realm of the user, and the users.deny file does not contain the user's UPN or a group the user belongs to.

4. If the users.allow file is empty or does not exist, and the users.deny file does not deny the user in anyway- explicitly, by the group, or by the realm.

5. If the users.allow file does not exist, and the users.deny file does not exist.

In all other cases, the user is denied access. Note that an empty users.allow file will be treated the same as if the users.allow file did not exist- the same applies to the users.deny file.

In all cases, the users.allow file takes precedence over the users.deny file. For example, if a user is group allowed but group denied, the user is granted access. Also, if a user is explicitly allowed, but explicitly denied by having the user's UPN in both files, then the user is granted access.

Note that in determining whether a given user is a member of a listed group, both the explicit group membership list of the given group and the user's primary group id are used. So if a user is not explicitly listed in a group's membership list, but a user's primary group ID is the same as the listed group's, then the user is considered a member of that group.

Disconnected Authentication

By default, pam_vas supports disconnected authentication. Each time a user successfully logs in against the Active Directory server, their password is stored as an SHA-1 hash in a secure cache file that is only readable by the root user. A timestamp is also stored with the hashed password, and cached passwords are good for 30 days.

This disconnected authentication allows services to continue to authenticate users if the network connection to the Active Directory server goes down, or if a laptop is used without plugging into the network. This behavior can be disabled by using the `no_disconnected` flag- in this case passwords will not be cached so no disconnected authentication can take place.

See Also

vastool (1), vascd (1)

D telnet Man Pages

Note: For the most update version of the man pages, check the product CD.

telnet (1)

telnet (1)

NAME

telnet

user interface to the TELNET protocol.

SYNOPSIS

`telnet [-78EFKLacdfrx] [-S tos] [-X authtype] [-e escape-char] [-k realm] [-l user] [-n tracefile] [host]telnet`

Description

The `telnet` command is used to communicate with another host using the TELNET protocol. If `telnet` is invoked without the host argument, it enters command mode, indicated by its prompt (telnet>). In this mode, it accepts and executes the commands listed below. If it is invoked with arguments, it performs an open command with those arguments.

Options:

`-8`	Specifies an 8-bit data path. This causes an attempt to negotiate the TELNET BINARY option on both input and output.
`-7`	Do not try to negotiate TELNET BINARY option.

`-E`	Stops any character from being recognized as an escape character.
`-F`	If Kerberos V5 authentication is being used, the -F option allows the local credentials to be forwarded to the remote system, including any credentials that have already been forwarded into the local environment.
`-K`	Specifies no automatic login to the remote system.
`-L`	Specifies an 8-bit data path on output. This causes the BINARY option to be negotiated on output.
`-S` *tos*	Sets the IP type-of-service (TOS) option for the telnet connection to the value *tos*, which can be a numeric TOS value or, on systems that support it, a symbolic TOS name found in the /etc/iptos file.
`-X` *atype*	Disables the *atype* type of authentication.
`-a`	Attempt automatic login. Currently, this sends the user name via the USER variable of the ENVIRON option if supported by the remote system. The name used is that of the current user as returned by getlogin(2) if it agrees with the current user ID, otherwise it is the name associated with the user ID.
`-c`	Disables the reading of the user's `.telnetrc` file. (See the toggle skiprc command on this man page.)
`-d`	Sets the initial value of the debug toggle to TRUE
`-e` *escape char*	Sets the initial telnet escape character to *escape char*. If *escape char* is omitted, then there will be no escape character.
`-f`	If Kerberos V5 authentication is being used, the `-f` option allows the local credentials to be forwarded to the remote system.

`-k` *realm*	If Kerberos authentication is being used, the `-k` option requests that telnet obtain tickets for the remote host in realm *realm* instead of the remote host's realm, as determined by krb_realmofhost(3).
`-l` *user*	When connecting to the remote system, if the remote system understands the ENVIRON option, then *user* will be sent to the remote system as the value for the variable USER. This option implies the `-a` option. This option may also be used with the open command.
`-n` *tracefile*	Opens *tracefile* for recording trace information. See the set tracefile command below.
`-r`	Specifies a user interface similar to rlogin(1). In this mode, the escape character is set to the tilde (~) character, unless modified by the `-e` option.
`-x`	Turn on encryption of the data stream. When this option is turned on, will exit with an error if authentication cannot be negotiated or if encryption cannot be turned on.
host	Indicates the official name, an alias, or the Internet address of a remote host.
port	Indicates a port number (address of an application). If a number is not specified, the default telnet port is used.

When in rlogin mode, a line of the form ~. disconnects from the remote host; ~ is the telnet escape character. Similarly, the line ~^Z suspends the telnet session. The line ~^] escapes to the normal telnet escape prompt.

Once a connection has been opened, telnet will attempt to enable the TELNET LINEMODE option. If this fails, then telnet will revert to one of two input modes: either "character at a time" or "old line by line" depending on what the remote system supports.

When LINEMODE is enabled, character processing is done on the local system, under the control of the remote system. When input editing or character echoing is to be disabled, the remote system will relay that information. The remote system will also relay changes to any special characters that happen on the remote system, so that they can take effect on the local system.

In "character at a time" mode, most text typed is immediately sent to the remote host for processing.

In "old line by line" mode, all text is echoed locally, and (normally) only completed lines are sent to the remote host. The "local echo character" (initially "^E") may be used to turn off and on the local echo (this would mostly be used to enter passwords without the password being echoed).

If the LINEMODE option is enabled, or if the localchars toggle is TRUE (the default for "old line by line"; see below), the user's quit, intr, and flush characters are trapped locally, and sent as TELNET protocol sequences to the remote side. If LINEMODE has ever been enabled, then the user's susp and eof are also sent as TELNET protocol sequences, and quit is sent as a TELNET ABORT instead of BREAK. There are options (see toggle autoflush and toggle autosynch below) which cause this action to flush subsequent output to the terminal (until the remote host acknowledges the TELNET sequence) and flush previous terminal input (in the case of quit and intr).

While connected to a remote host, telnet command mode may be entered by typing the telnet "escape character" (initially "^]"). When in command mode, the normal terminal editing conventions are available.

The following telnet commands are available. Only enough of each command to uniquely identify it need be typed (this is also true for arguments to the mode, set, toggle, unset, slc, environ, and display commands).

auth *argument* ... The `auth` command manipulates the information sent through the TELNET AUTHENTICATE option. Valid arguments for the auth command are as follows:

disable *type* Disables the specified type of authentication. To obtain a list of available types, use the auth disable ? command.

enable *type*
: Enables the specified type of authentication. To obtain a list of available types, use the auth enable ? command.

status
: Lists the current status of the various types of authentication.

close
: Close a TELNET session and return to command mode.

display *argument* ...
: Displays all, or some, of the set and toggle values (see below).

encrypt *argument* ...
: The encrypt command manipulates the information sent through the TELNET ENCRYPT option.

Note: Because of export controls, the TELNET ENCRYPT option is not supported outside of the United States and Canada.

Valid arguments for the encrypt command are as follows:

disable *type* [input | output]
: Disables the specified type of encryption. If you omit the input and output, both input and output are disabled. To obtain a list of available types, use the encrypt disable ? command.

enable *type* [input | output]
: Enables the specified type of encryption. If you omit input and output, both input and output are enabled. To obtain a list of available types, use the encrypt enable ? command.

input
: This is the same as the encrypt start input command.

-input
: This is the same as the encrypt stop input command.

output	This is the same as the encrypt start output command.
-output	This is the same as the encrypt stop output command.
start [input | output]	Attempts to start encryption. If you omit input and output, both input and output are enabled. To obtain a list of available types, use the encrypt enable ? command.
status	Lists the current status of encryption.
stop [input |output]	Stops encryption. If you omit input and output, encryption is on both input and output.
type *type*	Sets the default type of encryption to be used with later encrypt start or encrypt stop commands.

environ *arguments* ... The environ command is used to manipulate the variables that my be sent through the TELNET ENVIRON option. The initial set of variables is taken from the users environment, with only the DISPLAY and PRINTER variables being exported by default. The USER variable is also exported if the -a or -l options are used.

Valid arguments for the environ command are:

define *variable value*	Define the variable *variable* to have a value of *value*. Any variables defined by this command are automatically exported. The *value* may be enclosed in single or double quotes so that tabs and spaces may be included.
undefine *variable*	Remove *variable* from the list of environment variables.

	export *variable*	Mark the variable *variable* to be exported to the remote side.
	list	List the current set of environment variables. Those marked with a * will be sent automatically, other variables will only be sent if explicitly requested.
	?	Prints out help information for the environ command.

logout

Sends the TELNET LOGOUT option to the remote side. This command is similar to a close command; however, if the remote side does not support the LOGOUT option, nothing happens. If, however, the remote side does support the LOGOUT option, this command should cause the remote side to close the TELNET connection. If the remote side also supports the concept of suspending a user's session for later reattachment, the logout argument indicates that you should terminate the session immediately.

mode *type*

Type is one of several options, depending on the state of the TELNET session. The remote host is asked for permission to go into the requested mode. If the remote host is capable of entering that mode, the requested mode will be entered.

character	Disable the TELNET LINEMODE option, or, if the remote side does not understand the LINEMODE option, then enter "character at a time" mode.
line	Enable the TELNET LINEMODE option, or, if the remote side does not understand the LINEMODE option, then attempt to enter "old-line-by-line" mode.
isig (-isig)	Attempt to enable (disable) the TRAPSIG mode of the LINEMODE option. This

		requires that the LINEMODE option be enabled.
	edit (-edit)	Attempt to enable (disable) the EDIT mode of the LINEMODE option. This requires that the LINEMODE option be enabled.
	softtabs (-softtabs)	Attempt to enable (disable) the SOFT_TAB mode of the LINEMODE option. This requires that the LINEMODE option be enabled.
	litecho (-litecho)	Attempt to enable (disable) the LIT_ECHO mode of the LINEMODE option. This requires that the LINEMODE option be enabled.
	?	Prints out help information for the mode command.
open *host* [-l *user*]	Open a connection to the named host. If no port number is specified, telnet will attempt to contact a TELNET server at the default port. The host specification may be either a host name (see hosts(5)) or an Internet address specified in the "dot notation" (see inet(3)). The `[-l]` option may be used to specify the user name to be passed to the remote system via the ENVIRON option. When connecting to a non-standard port, telnet omits any automatic initiation of TELNET options. When the port number is preceded by a minus sign, the initial option negotiation is done. After establishing a connection, the file `.telnetrc` in the users home directory is opened. Lines beginning with a # are comment lines. Blank lines are ignored. Lines that begin without white space are the start of a machine entry. The first thing on the line is the name of the machine that is being connected to. The rest of the line, and successive lines that begin with white space are assumed to be telnet commands and are processed as if they had been typed in manually to the telnet command prompt.	

quit — Close any open TELNET session and exit telnet. An end of file (in command mode) will also close a session and exit.

send *arguments* — Sends one or more special character sequences to the remote host. The following are the arguments which may be specified (more than one argument may be specified at a time):

abort	Sends the TELNET ABORT (Abort processes) sequence.
ao	Sends the TELNET AO (Abort Output) sequence, which should cause the remote system to flush all output from the remote system to the user's terminal.
ayt	Sends the TELNET AYT (Are You There) sequence, to which the remote system may or may not choose to respond.
brk	Sends the TELNET BRK (BReaK) sequence, which may have significance to the remote system.
ec	Sends the TELNET EC (Erase Character) sequence, which should cause the remote system to erase the last character entered.
el	Sends the TELNET EL (Erase Line) sequence, which should cause the remote system to erase the line currently being entered.
eof	Sends the TELNET EOF (End Of File) sequence.
eor	Sends the TELNET EOR (End Of Record) sequence.
escape	Sends the current telnet escape character

	(initially "^").
ga	Sends the TELNET GA (Go Ahead) sequence, which likely has no significance to the remote system.
getstatus	If the remote side supports the TELNET STATUS command, getstatus will send the subnegotiation to request that the server send its current option status.
ip	Sends the TELNET IP (Interrupt Process) sequence, which should cause the remote system to abort the currently running process.
nop	Sends the TELNET NOP (No OPeration) sequence.
susp	Sends the TELNET SUSP (SUSPend process) sequence.
synch	Sends the TELNET SYNCH sequence. This sequence causes the remote system to discard all previously typed (but not yet read) input. This sequence is sent as TCP urgent data (and may not work if the remote system is a 4.2BSD system -- if it doesn't work, a lower case "r" may be echoed on the terminal).
do *cmd*	
dont *cmd*	
will *cmd*	
wont *cmd*	Sends the TELNET DO cmd sequence. Cmd can be either a decimal number

between 0 and 255, or a symbolic name for a specific TELNET command. Cmd can also be either help or ? to print out help information, including a list of known symbolic names.

? — Prints out help information for the send command.

set *argument value*

.unset *argument value*

The set command will set any one of a number of telnet variables to a specific value or to TRUE. The special value off turns off the function associated with the variable, this is equivalent to using the unset command. The unset command will disable or set to FALSE any of the specified functions. The values of variables may be interrogated with the display command. The variables which may be set or unset, but not toggled, are listed here. In addition, any of the variables for the toggle command may be explicitly set or unset using the set and unset commands.

ayt — If TELNET is in localchars mode, or LINEMODE is enabled, and the status character is typed, a TELNET AYT sequence (see send ayt preceding) is sent to the remote host. The initial value for the "Are You There" character is the terminal's status character.

echo — This is the value (initially "^E") which, when in "line by line" mode, toggles between doing local echoing of entered characters (for normal processing), and suppressing echoing of entered characters (for entering, say, a password).

eof	If telnet is operating in LINEMODE or "old line by line" mode, entering this character as the first character on a line will cause this character to be sent to the remote system. The initial value of the eof character is taken to be the terminal's eof character.
erase	If telnet is in localchars mode (see toggle localchars below), and if telnet is operating in "character at a time" mode, then when this character is typed, a TELNET EC sequence (see send ec above) is sent to the remote system. The initial value for the erase character is taken to be the terminal's erase character.
escape	This is the telnet escape character (initially "^[") which causes entry into telnet command mode (when connected to a remote system).
flushoutput	If telnet is in localchars mode (see toggle localchars below) and the flushoutput character is typed, a TELNET AO sequence (see send ao above) is sent to the remote host. The initial value for the flush character is taken to be the terminal's flush character.
forw1	
forw2	If TELNET is operating in LINEMODE, these are the characters that, when typed, cause partial lines to be forwarded to the remote system. The initial value for the forwarding characters are taken from the terminal's eol and eol2 characters.

interrupt	If telnet is in localchars mode (see toggle localchars below) and the interrupt character is typed, a TELNET IP sequence (see send ip above) is sent to the remote host. The initial value for the interrupt character is taken to be the terminal's intr character.
kill	If telnet is in localchars mode (see toggle localchars below), and if telnet is operating in "character at a time" mode, then when this character is typed, a TELNET EL sequence (see send el above) is sent to the remote system. The initial value for the kill character is taken to be the terminal's kill character.
lnext	If telnet is operating in LINEMODE or "old line by line" mode, then this character is taken to be the terminal's lnext character. The initial value for the lnext character is taken to be the terminal's lnext character.
quit	If telnet is in localchars mode (see toggle localchars below) and the quit character is typed, a TELNET BRK sequence (see send brk above) is sent to the remote host. The initial value for the quit character is taken to be the terminal's quit character.
reprint	If telnet is operating in LINEMODE or "old line by line" mode, then this character is taken to be the terminal's reprint character. The initial value for the reprint character is taken to be the terminal's reprint character.
rlogin	This is the rlogin escape character. If set,

	the normal TELNET escape character is ignored unless it is preceded by this character at the beginning of a line. This character, at the beginning of a line followed by a "." closes the connection; when followed by a ^Z it suspends the telnet command. The initial state is to disable the rlogin escape character.
start	If the TELNET TOGGLE-FLOW-CONTROL option has been enabled, then this character is taken to be the terminal's start character. The initial value for the kill character is taken to be the terminal's start character.
stop	If the TELNET TOGGLE-FLOW-CONTROL option has been enabled, then this character is taken to be the terminal's stop character. The initial value for the kill character is taken to be the terminal's stop character.
susp	If telnet is in localchars mode, or LINEMODE is enabled, and the suspend character is typed, a TELNET SUSP sequence (see send susp above) is sent to the remote host. The initial value for the suspend character is taken to be the terminal's suspend character.
tracefile	This is the file to which the output, caused by netdata or option tracing being TRUE, will be written. If it is set to "-", then tracing information will be written to standard output (the default).
worderase	If telnet is operating in LINEMODE or

"old line by line" mode, then this character is taken to be the terminal's worderase character. The initial value for the worderase character is taken to be the terminal's worderase character.

? — Displays the legal set (unset) commands.

slc *state*

The slc command (Set Local Characters) is used to set or change the state of the special characters when the TELNET LINEMODE option has been enabled. Special characters are characters that get mapped to TELNET commands sequences (like ip or quit) or line editing characters (like erase and kill). By default, the local special characters are exported.

check — Verify the current settings for the current special characters. The remote side is requested to send all the current special character settings, and if there are any discrepancies with the local side, the local side will switch to the remote value.

export — Switch to the local defaults for the special characters. The local default characters are those of the local terminal at the time when telnet was started.

import — Switch to the remote defaults for the special characters. The remote default characters are those of the remote system at the time when the TELNET connection was established.

? — Prints out help information for the slc command.

status

Show the current status of telnet. This includes the peer one is connected to, as well as the current mode.

toggle *arguments ...* Toggle (between TRUE and FALSE) various flags that control how telnet responds to events. These flags may be set explicitly to TRUE or FALSE using the set and unset commands listed above. More than one argument may be specified. The state of these flags may be interrogated with the display command. Valid arguments are:

authdebug	Turns on debugging information for the authentication code.
autoflush	If autoflush and localchars are both TRUE, then when the ao, or quit characters are recognized (and transformed into TELNET sequences; see set above for details), telnet refuses to display any data on the user's terminal until the remote system acknowledges (via a TELNET TIMING MARK option) that it has processed those TELNET sequences. The initial value for this toggle is TRUE if the terminal user had not done an "stty noflsh", otherwise FALSE (see stty(1)).
autodecrypt	When the TELNET ENCRYPT option is negotiated, by default the actual encryption (decryption) of the data stream does not start automatically. The autoencrypt (autodecrypt) command states that encryption of the output (input) stream should be enabled as soon as possible. Note: Because of export controls, the TELNET ENCRYPT option is not supported outside the United States and Canada.
autologin	If the remote side supports the TELNET AUTHENTICATION option TELNET

attempts to use it to perform automatic authentication. If the AUTHENTICATION option is not supported, the user's login name are propagated through the TELNET ENVIRON option. This command is the same as specifying a option on the open command.

autosynch — If autosynch and localchars are both TRUE, then when either the intr or quit characters is typed (see set above for descriptions of the intr and quit characters), the resulting TELNET sequence sent is followed by the TELNET SYNCH sequence. This procedure should cause the remote system to begin throwing away all previously typed input until both of the TELNET sequences have been read and acted upon. The initial value of this toggle is FALSE.

binary — Enable or disable the TELNET BINARY option on both input and output.

inbinary — Enable or disable the TELNET BINARY option on input.

outbinary — Enable or disable the TELNET BINARY option on output.

crlf — If this is TRUE, then carriage returns will be sent as CR LF. If this is FALSE, then carriage returns will be send as CR NUL. The initial value for this toggle is FALSE.

crmod — Toggle carriage return mode. When this mode is enabled, most carriage return characters received from the remote host

	will be mapped into a carriage return followed by a line feed. This mode does not affect those characters typed by the user, only those received from the remote host. This mode is not very useful unless the remote host only sends carriage return, but never line feed. The initial value for this toggle is FALSE.
debug	Toggles socket level debugging (useful only to the super user). The initial value for this toggle is FALSE.
encdebug	Turns on debugging information for the encryption code.
localchars	If this is TRUE, then the flush, interrupt, quit, erase, and kill characters (see set above) are recognized locally, and transformed into (hopefully) appropriate TELNET control sequences (respectively ao, ip, brk, ec, and el; see send above). The initial value for this toggle is TRUE in "old line by line" mode, and FALSE in "character at a time" mode. When the LINEMODE option is enabled, the value of localchars is ignored, and assumed to always be TRUE. If LINEMODE has ever been enabled, then quit is sent as abort, and eof and suspend are sent as eof and susp, see send above).
netdata	Toggles the display of all network data (in hexadecimal format). The initial value for this toggle is FALSE.
options	Toggles the display of some internal telnet protocol processing (having to do with

		TELNET options). The initial value for this toggle is FALSE.
	prettydump	When the netdata toggle is enabled, if prettydump is enabled the output from the netdata command will be formatted in a more user readable format. Spaces are put between each character in the output, and the beginning of any TELNET escape sequence is preceded by a '*' to aid in locating them.
	skiprc	When the skiprc toggle is TRUE, TELNET skips the reading of the `.telnetrc` file in the users home directory when connections are opened. The initial value for this toggle is FALSE.
	termdata	Toggles the display of all terminal data (in hexadecimal format). The initial value for this toggle is FALSE.
	verbose_encrypt	When the verbose_encrypt toggle is TRUE, TELNET prints out a message each time encryption is enabled or disabled. The initial value for this toggle is FALSE. Note: Because of export controls, data encryption is not supported outside of the United States and Canada.
	?	Displays the legal toggle commands.
z		Suspend telnet. This command only works when the user is using the csh(1).
! *[command]*		Execute a single command in a subshell on the local system. If command is omitted, then an interactive subshell is invoked.
? *[command]*		Get help. With no arguments, telnet prints a help summary. If a

command is specified, telnet will print the help information for just that command.

environment

Telnet uses at least the HOME, SHELL, DISPLAY, and TERM environment variables. Other environment variables may be propagated to the other side via the TELNET ENVIRON option.

files

`~/.telnetrc` user customized telnet startup values

history

The Telnet command appeared in 4.2BSD.

notes

On some remote systems, echo has to be turned off manually when in "old line by line" mode.

In "old line by line" mode or LINEMODE the terminal's eof character is only recognized (and sent to the remote system) when it is the first character on a line.

telnetd(8) Man Page

telnetd (8)

NAME

telnetd

DARPA TELNET protocol server

SYNOPSIS

`telnetd [-BUhkln] [-D debugmode] [-S tos] [-X authtype] [-a authmode] [-rlowpty-highpty] [-u len] [-debug] [-L /bin/login] [-y port]telnetd`

DESCRIPTION

The `telnetd` is a server which supports the DARPA standard TELNET virtual terminal protocol. Telnetd is normally invoked by the internet server (see inetd(8)) for requests to connect to the TELNET port as indicated by the `/etc/services` file (see services(5)). The -debug option maybe used to start up telnetd manually, instead of through inetd(8). If started up this way, *port* may be specified to run telnetd on an alternate TCP port number.

The telnetd command accepts the following options:

`-a` *`authmode`* This option may be used for specifying what mode should be used for authentication. Note that this option is only useful if telnetd has been compiled with support for the AUTHENTICATION option. There are several valid values for *authmode*:

`debug`	Turns on authentication debugging code.
`user`	Only allow connections when the remote user can provide valid authentication information to identify the remote user, and is allowed access to the specified account without providing a password.
`valid`	Only allow connections when the remote user can provide valid authentication information to identify the remote user. The login(1) command will provide any additional user verification needed if the remote user is not allowed automatic access to the specified account.
`other`	Only allow connections that supply some authentication information. This option is currently not supported by any of the

existing authentication mechanisms, and is thus the same as specifying -a valid.

`otp` — Only allow authenticated connections (as with -a user) and also logins with one-time passwords (OTPs). This option will call login with an option so that only OTPs are accepted. The user can of course still type secret information at the prompt.

`none` — This is the default state. Authentication information is not required. If no or insufficient authentication information is provided, then the login(1) program will provide the necessary user verification.

`off` — This disables the authentication code. All user verification will happen through the login(1) program.

`-B` — Ignored.

`-D` *degubmode* — This option may be used for debugging purposes. This allows telnetd to print out debugging information to the connection, allowing the user to see what telnetd is doing. There are several possible values for *debugmode*:

`options` — Prints information about the negotiation of TELNET options.

`report` — Prints the options information, plus some additional information about what processing is going on.

`netdata` — Displays the data stream received by telnetd.

`ptydata` — Displays data written to the pty.

`exercise` Has not been implemented yet.

`-h` Disables the printing of host-specific information before login has been completed.

`-k`

`-l` Ignored.

`-n` Disable TCP keep-alives. Normally telnetd enables the TCP keep-alive mechanism to probe connections that have been idle for some period of time to determine if the client is still there, so that idle connections from machines that have crashed or can no longer be reached may be cleaned up.

`-r` *`lowpty-highpty`*

This option is only enabled when telnetd is compiled for UNICOS. It specifies an inclusive range of pseudo-terminal devices to use. If the system has sysconf variable _SC_CRAY_NPTY configured, the default pty search range is 0 to _SC_CRAY_NPTY; otherwise, the default range is 0 to 128. Either *lowpty* or *highpty* may be omitted to allow changing either end of the search range. If *lowpty* is omitted, the - character is still required so that telnetd can differentiate *highpty* from *lowpty*.

`-S` *`tos`*

`-u` *`len`* This option is used to specify the size of the field in the utmp structure that holds the remote host name. If the resolved host name is longer than *len*, the dotted decimal value will be used instead. This allows hosts with very long host names that overflow this field to still be uniquely identified. Specifying -u0 indicates that only dotted decimal addresses should be put into the *utmp* file.

`-U`	This option causes telnetd to refuse connections from addresses that cannot be mapped back into a symbolic name via the gethostbyaddr(3) routine.
`-X` *`authtype`*	This option is only valid if telnetd has been built with support for the authentication option. It disables the use of *authtype* authentication, and can be used to temporarily disable a specific authentication type without having to recompile telnetd.
`-L` *`pathname`*	Specify pathname to an alternative login program.
`-y`	Makes telnetd not warn when a user is trying to login with a cleartext password.

Telnetd operates by allocating a pseudo-terminal device (see pty(4)) for a client, then creating a login process which has the slave side of the pseudo-terminal as stdin, stdout and stderr. Telnetd manipulates the master side of the pseudo-terminal, implementing the TELNET protocol and passing characters between the remote client and the login process.

When a TELNET session is started up, telnetd sends TELNET options to the client side indicating a willingness to do the following TELNET options, which are described in more detail below:

```
DO AUTHENTICATION

WILL ENCRYPT

DO TERMINAL TYPE

DO TSPEED

DO XDISPLOC

DO NEW-ENVIRON

DO ENVIRON
```

```
WILL SUPRESS GO AHEAD

DO ECHO

DO LINEMODE

DO NAWS

WILL STATUS

DO LFLOW

DO TIMING-MARK
```

The pseudo-terminal allocated to the client is configured to operate in "cooked" mode, and with XTABS and CRMOD enabled (see tty(4)).

Telnetd has support for enabling locally the following TELNET options:

WILL ECHO	When the LINEMODE option is enabled, a WILL ECHO or WONT ECHO will be sent to the client to indicate the current state of terminal echoing. When terminal echo is not desired, a WILL ECHO is sent to indicate that telnetd will take care of echoing any data that needs to be echoed to the terminal, and then nothing is echoed. When terminal echo is desired, a WONT ECHO is sent to indicate that telnetd will not be doing any terminal echoing, so the client should do any terminal echoing that is needed.
WILL BINARY	Indicates that the client is willing to send a 8 bits of data, rather than the normal 7 bits of the Network Virtual Terminal.
WILL SGA	Indicates that it will not be sending IAC GA, go ahead, commands.
WILL STATUS	Indicates a willingness to send the client, upon request, of the current status of all TELNET options.

WILL TIMING-MARK

Whenever a DO TIMING-MARK command is received, it is always responded to with a WILL TIMING-MARK

WILL LOGOUT — When a DO LOGOUT is received, a WILL LOGOUT is sent in response, and the TELNET session is shut down.

WILL ENCRYPT — Only sent if telnetd is compiled with support for data encryption, and indicates a willingness to decrypt the data stream.

Telnetd has support for enabling remotely the following TELNET options:

DO BINARY — Sent to indicate that telnetd is willing to receive an 8 bit data stream.

DO LFLOW — Requests that the client handle flow control characters remotely.

DO ECHO — This is not really supported, but is sent to identify a 4.2BSD telnet(1) client, which will improperly respond with WILL ECHO. If a WILL ECHO is received, a DONT ECHO will be sent in response.

DO TERMINAL-TYPE

Indicates a desire to be able to request the name of the type of terminal that is attached to the client side of the connection.

DO SGA — Indicates that it does not need to receive IAC GA, the go ahead command.

DO NAWS — Requests that the client inform the server when the window (display) size changes.

DO TERMINAL-SPEED

Indicates a desire to be able to request information about the speed of the serial line to which the client is attached.

DO XDISPLOC — Indicates a desire to be able to request the name of the X windows display that is associated with the telnet client.

DO NEW-ENVIRON

Indicates a desire to be able to request environment variable information, as described in RFC 1572.

DO ENVIRON — Indicates a desire to be able to request environment variable information, as described in RFC 1408.

DO LINEMODE — Only sent if telnetd is compiled with support for linemode, and requests that the client do line by line processing.

DO TIMING-MARK

Only sent if telnetd is compiled with support for both linemode and kludge linemode, and the client responded with WONT LINEMODE. If the client responds with WILL TM, the it is assumed that the client supports kludge linemode. Note that the [-k] option can be used to disable this.

DO AUTHENTICATION

Only sent if telnetd is compiled with support for authentication, and indicates a willingness to receive authentication information for automatic login.

DO ENCRYPT — Only sent if telnetd is compiled with support for data encryption, and indicates a willingness to decrypt the data stream.

FILES

/etc/services

/etc/inittab (UNICOS systems only)

/etc/iptos (if supported)

SEE ALSO

telnet(1), login(1)

STANDARDS

RFC-854	TELNET PROTOCOL SPECIFICATION
RFC-855	TELNET OPTION SPECIFICATIONS
RFC-856	TELNET BINARY TRANSMISSION
RFC-857	TELNET ECHO OPTION
RFC-858	TELNET SUPPRESS GO AHEAD OPTION
RFC-859	TELNET STATUS OPTION
RFC-860	TELNET TIMING MARK OPTION
RFC-861	TELNET EXTENDED OPTIONS - LIST OPTION
RFC-885	TELNET END OF RECORD OPTION
RFC-1075	Telnet Window Size Option
RFC-1079	Telnet Terminal Speed Option
RFC-1091	Telnet Terminal-Type Option
RFC-1096	Telnet X Display Location Option
RFC-1123	Requirements for Internet Hosts -- Application and Support
RFC-1184	Telnet Linemode Option
RFC-1372	Telnet Remote Flow Control Option
RFC-1416	Telnet Authentication Option

RFC-1411	Telnet Authentication: Kerberos Version 4
RFC-1412	Telnet Authentication: SPX
RFC-1571	Telnet Environment Option Interoperability Issues
RFC-1572	Telnet Environment Option

BUGS

Some TELNET commands are only partially implemented.

Because of bugs in the original 4.2 BSD telnet(1), telnetd performs some dubious protocol exchanges to try to discover if the remote client is, in fact, a 4.2 BSD telnet(1).

Binary mode has no common interpretation except between similar operating systems (Unix in this case).

The terminal type name received from the remote client is converted to lower case.

Telnetd never sends TELNET IAC GA (go ahead) commands.

E vascd(1) Man Page

Note: For the most update version of the man pages, check the product CD.

vascd (1)

NAME

vascd

The SCO Authentication client daemon

SYNOPSIS

`vascd [-h] [-v] [-l] [-d] [-p pidfile] [-o user] [-f log-file] [-c update-interval] [-n principal] [-t timesync-inteval] [-r realmscache-sync-interval]vascd`

Description

vascd is a daemon that must be started on UNIX/Linux workstations in order for SCO Authentication to operate correctly. When started, vascd authenticates to Active Directory using credentials that were established at the time that the computer object was created in the Active Directory domain (see vastool (1) for details). vascd then uses this secure connection to Active Directory to proxy and cache user and group account information for other processes.

The use of vascd allows several important features:

Security - Due to the way that PAM and NSS subsystems operate, most LDAP based UNIX account management solutions require that anonymous or public access be allowed to UNIX account attributes. Since vascd authenticates as an Active Directory domain computer, vascd can access UNIX account information that is protected by Active Directory access control restrictions.

Scalability - Due to the way that the PAM and NSS subsystems operate, most LDAP based UNIX account management solutions generate excessive numbers of LDAP con-

nections and LDAP search requests. This results in dramatically increased network traffic and load on the LDAP server. vascd establishes a single connection to Active Directory for the computer it is running on, and proxies all of the NSS requests through that single connection. At the same time, vascd is able to perform intelligent caching of frequently used information so that LDAP traffic is reduced to the absolute minimum.

Disconnected Operation - Since vascd maintains a persistent cache of frequently used information, it is possible for the entire SCO Authentication system to continue to operate in environments where the network connection to Active Directory is unreliable or completely unavailable.

vascd Options

These are the options you can pass to `vascd` when starting the daemon.

`-h`	Shows the `vascd` usage.
`-v`	Print the `vascd` version and exit.
`-l`	Print the `vascd` licensing information and exit.
`-d`	Causes the daemon to not detach from the controlling tty, and prints debugging messages to stdout. This is useful for debugging purposes.
`-p pidfile`	Specify an alternative pid file. Default is `/var/state/vas/vascd/.vascd.pid`
`-o user`	Run vascd as the specified user. Default is daemon.
`-f logfile`	Specify an alternative log file. Default uses syslog if the system supports syslog. Otherwise, vascd will log to `/var/log/vascd` by default.

`-c update-interval`

Specify the "blackout" period in seconds. Default is 600 seconds (10 minutes). This setting can also be set in `vas.conf` file in the "[vascd]" section. An example is:

[vascd] update-interval = <seconds>

In order to minimize network traffic and the load on the directory server, vascd aggressively caches data that is retrieved from the directory server so that subsequent requests can be satisfied from the cache with out having to contact the directory server. The cache is only updated when it is determined that a change has been made in the directory. Nevertheless, due to the way that the NSS subsystem is called it is not uncommon for hundreds of requests to be generated in a matter of seconds. Therefore, in order to further reduce the load on the network and on the directory, vascd enforces a "blackout period" during which all requests will be resolved from the cache.

By default the blackout period is set to 10 minutes. What this means is that the addition of new users and changes to UNIX account information may take as long as 10 minutes to become visible on the Linux/Unix workstation due to the blackout period. For environments where changes must take affect quicker, it is possible to change the "blackout period" using the `-c` command line option when starting vascd. Using `-c` to decrease the "blackout period" will result in a system where hosts are more responsive to changes made in Active Directory-- at the expense of increased network traffic and load on the Active Directory server.

The amount of additional network traffic depends on the number of Linux/UNIX hosts and their use. In small installations (less than 100 hosts or less than 100 users) the "blackout period" could be safely reduced. In larger installations it is recommended that the "blackout period" be left at the default value or increased to 30 minutes or 1 hour. Regardless of the "blackout period", the

administrator can force vascd to update the cache immediately by signaling vascd with SIGHUP or by executing `vastool flush`, which will cause vascd to reload all of it's information at the next NSS request.

`-n principal` The principal name that vascd will authenticate as. There must be a valid keytab entry in the vas.keytab file for the principal. By default `vascd` will authenticate as the host principal created during `vastool join`.

`-t timesync-interval`

vascd will operate as a time synchronization agent if, on start up, vascd detects that no other process has bound the NTP port (123). If the NTP port is not bound, vascd will issue SNTP requests to the host that was configured with the `vastool join` or `vastool configure realm`. The `-t` option allows the system administrator to specify the frequency (in hours) that time synchronization occurs. The default is every 12 hours. If the timesync-interval is set to 0 vascd will not operate as a timesync agent.

If a specialized NTP daemon is bind to synchronize time it is crucial that this daemon be started *before* vascd. This way the NTP port will be bound when vascd starts. vascd will not operate as a timesync agent and there will be no conflicts. If, for some reason, vascd must be started before the NTP daemon, then the `-t` option should be set to zero to disable vascd timesync and avoid what would otherwise be time synchronization thrashing between the specialized NTP daemon and vascd.

This option can also be specified in the "[vascd]" section. An example is:

[vascd] timesync-interval = <hours>

`-r realmscache-sync-interval`

`vascd` will rebuild the realms cache periodically. This cache is used to reduce the amount of DNS traffic that is needed in order to discover all of the services in the Active Directory Domains- `vascd` will rebuild this cache periodically to get any updates that have been made. Setting this option to 0 will disable the realms cache syncing.

This option can also be specified in the "[vascd]" section. An example is:

[vascd] realmscache-sync-interval = <minutes>

See Also

vastool (1), pam_vas (5)

F vastool(1) Man Page

Note: For the most update version of the man pages, check the product CD.

vastool (1)

NAME

vastool

The command line administration tool for SCO Authentication.

SYNOPSIS

vastool [-h] [-v] [-u *username] [-w password] [-s]{vastool_command}* [*vastool_command arguments] [vastool options] attrs [-u] [-s] [-g] [-d]{objectname} {attribute-Name}* [vastool options] configure [vastool options] create [-g] [-c *container] [-p password] [-i info] [-x]{name}* [vastool options] delete [-g] [-n] [-d] name... [vastool options] flush [-r] [accounts | auth | keytab | users | groups] [vastool options] group group_name add | del name... [vastool options] join [-c *container] [-n computer_name] [-f] realm_name [servers...] [vastool options] kinit [service] klist [vastool options] kdestroy [vastool options] list [-a] [-l] [-f] [-c] users | user {username} | groups | group {groupname} [vastool options] load [-f file] [-c container] [-p password] [-x] users | groups [vastool options] merge [users | user {username} | groups] [vastool options] nis-import [-o] [-f filename] [-c container] [-s format_string] NIS Map Name [vastool options] passwd [-s] [-c] [user_name] [vastool options] realms find [root | services [realm] | gc] | cache {update | list} [vastool options] timesync [timeserver] [-q] [vastool options] unconfigure nss | pam [service...] [vastool options] unjoin [-n computer_name] [vastool options]*

*unmerge [users | groups] [vastool options] ypcat [-k] [-x] NIS Map Name*vastool

Description

vastool is a command line program that allows you to configure the SCO Authentication client, access information stored in Active Directory, and to store information in Active Directory. vastool is located at /opt/vas/bin/vastool. It has been designed to be script-friendly and to allow administrators to easily manage users, groups, and other information stored in Active Directory from UNIX/Linux workstations.

In order to run vastool, you must specify the options for vastool, a command to run, and the options for that specific command. The following is a list of supported vastool commands and a brief description of each command's purpose. A more detailed explanation of each command will follow later.

attrs	List an Active Directory object's attributes.
configure	Update PAM, NSS, and other configuration files to use the SCO Authentication components.
create	Create a User, Group, or Computer Object in Active Directory.
delete	Delete users, groups, computer objects, or NIS Map objects in Active Directory.
flush	Flush the vascd cache.
group	Add or remove users from Active Directory groups.
join	Configure the system to use pam_vas for authentication, nss_vas for NSS information for users and groups, adds a computer object to Active Directory, and starts vascd.
kinit	Perform kinit functions - obtains Kerberos ticket(s) for service(s).
klist	Perform klist functions - lists the Kerberos tickets stored in the

	calling user's credentials cache.
`kdestroy`	Perform kdestroy functions- destroys all existing tickets in the calling user's credentials cache.
`license`	Install a license for the installation.
`list`	List users and groups in Active Directory, along with their Unix account information.
`load`	Import users and groups into Active Directory from a file that follows the format of /etc/password or /etc/group.
`merge`	Merge Active Directory users and groups into /etc/password and /etc/group.
`nis-import`	Load NIS Maps into Active Directory.
`passwd`	Change your password or reset another user's password in Active Directory.
`realms`	Detect the realms (domains) on your network and the servers providing LDAP and Kerberos services for those realms.
`timesync`	Query and syncronize system time with Active Directory or other specified time server.
`unconfigure`	Update PAM, NSS, and other configuration files to not use the SCO Authentication components.
`unjoin`	Configure the system to not use the SCO Authentication client for authentication and for NSS, and then removes the computer object from Active Directory.
`unmerge`	Remove Active Directory users and groups from /etc/password and /etc/group.
`ypcat`	Provides functionality similar to ypcat- prints out the contents of NIS Map objects stored in Active Directory

Vastool Options

These are the options you can pass to `vastool`. They must be specified before the command name.

`-h [command]`	If no command is specified, it shows the `vastool` usage and a list of available commands. If a command is specified, it shows the usage for that vastool command.
`-v`	Print out the `vastool` version and exit.
`-u principal`	Sets the principal name to authenticate as when the vastool command needs to access Active Directory. If the caller has root access, "host/" can be specified and vastool will authenticate as the computer object that `vastool` is running on. If `-u` is not used, then `vastool` will authenticate as the calling user, and will attempt to reuse Kerberos tickets from the user's credentials cache. If `-u` is specified, then no existing credentials cache will be used, and new tickets obtained will not be saved to disk.
`-w password`	This option allows you pass in a password on the command line. Please note that this is a security hole in a production environment, and should never be used, except in testing environments.
`-s`	This option will cause `vastool` to not prompt for any initial passwords, but instead read them from stdin. This allows you to use some vastool commands in a non-interactive mode.

Vastool Command Synopsis

The following is a detailed description of all the available `vastool` commands. Their usage descriptions, a detailed explanation of their purpose, how they work, and examples are included for each command.

vastool attrs

`vastool attrs` can be used to view attributes stored on objects in Active Directory.

These attributes are the LDAP attributes on the objects in Active Directory.

`vastool`

vastool options

attrs

-u

-s

-g

-d

objectname

attributeName

The objectname parameter will be interpreted differently according to the flag used. The following list explains how each flag works.

`-u`	Interprets objectname as a user name. This is the default behavior if no flag is specified.
`-s`	Interprets objectname as a service principal name.
`-g`	Interprets objectname as a group name.
`-d`	Interprets objectname as an LDAP distinguished name. This allows you to view the attributes on any object in Active Directory.

`-u` will cause `vastool` to interpret the name as a user name. This is also the default behavior if no flags are specified. However, if the user name starts with "host/", then the name will be interpreted as a service principal name. `-s` will interpret the name as a Kerberos service principal name. `-g` will interpret the name as a group name.

Following is an example of getting the home directory for the user john, getting the last time the computers container was modified, and getting a list of the members of the eng group.

vastool attrs john unixHomeDirectory

vastool attrs -d "CN=computers,DC=example,DC=com" whenChanged

vastool attrs -g eng member

vastool configure

`vastool configure` can be used to modify your system's PAM, NSS, and your Kerberos realm configuration. This command must be run as root.

`vastool`

vastool options

configure

realm realm_name [servers...] | extra-realm realm_name [servers...] | computer-name name | samba [smb.conf path] [workgroup] | nss | pam [service...]

Note that most of these command are for advanced usage only. The `vastool join` will automatically perform these steps for you.

Configuring the realm will modify `/etc/opt/vas/vas.conf` to use the given realm_name as your default realm. If a list of server names is passed in, these servers will stored as the servers for the given realm. In Active Directory terms, the realm will be the domain name of the domain this computer will be a member of. `vastool configure extra-realm` can also be used to configure other domains if you need to support multiple servers in your Active Directory tree. This will add information for these realms, but it will not make the new realm the default realm.

Configuring samba will integrate your local samba configuration with the SCO Authentication configuration. You will be prompted for the location of your smb.conf file and your workgroup/domain (these can optionally be specified as arguments on the command line), then the smb.conf will be updated to work with Active Directory and SCO

Authentication. Samba's password for authenticating as the computer will be synchronized with the SCO Authentication computer password. This allows your Samba server to run with domain security alongside SCO Authentication on the same machine.

If you ever reset the computer password, then you will need to run the `vastool configure samba` to synchronize the computer password between Samba and SCO Authentication again, otherwise Samba will no longer be able to authenticate as the computer.

Configuring NSS will modify the passwd and group entries in the `/etc/nsswitch.conf` to include an entry for `vas`, (the SCO Authentication NSS module), after the `files` NSS module. You can manually edit `/etc/nsswitch.conf` to put `vas` in front of `files`. This change will cause any username that has both a local account and a SCO Authentication account to be resolved to the SCO Authentication account. By default, local accounts will override Active Directory accounts. At this time, no other NSS subsystems besides passwd and group are supported.

Configuring PAM will modify either `/etc/pam.conf`, or the files located in `/etc/pam.d/` to use the `pam_vas` PAM module. If no service names are specified, then all existing services, including the default "other", will be configured to use pam_vas. For more information on configuring and customizing pam_vas, see pam_vas (5).

If you are configuring a computer whose hostname will not always match up with the name of the computer object in Active Directory that represents your computer, you must use `vastool configure computer-name` to specify the name of the computer as it appears in Active Directory. Otherwise, your computer will not be able to communicate with Active Directory. When using `vastool join` you can specify this name with the `-n` option.

Following is an example configuring the example.com realm, configuring the example.com realm and using a special name for the host computer, configuring NSS, configuring all PAM enabled services, and configuring the login, telnet, and ssh services to use SCO Authentication.

vastool configure realm example.com

vastool configure extra-realm sub.example.com server.sub.example.com

vastool configure computer-name mycomputer

vastool configure nss

vastool configure pam

vastool configure pam login telnet sshd

If you are configuring a Solaris 8 or 9 system, you will need to use the `-g` option so that vastool will configure the pam.conf file to better handle authentication. The SCO Authentication module will prompt all users for their passwords instead other modules. This allows SCO Authentication to do better checking for authentication and improve security. Using this option causes vastool to modify the pam.conf file by either commenting out, altering, adding, or removing pam options for other modules in the file besides those for the SCO Authentication line entries.

vastool configure pam -g

vastool configure pam -g login telnet sshd

vastool create

`vastool create` can be used to create a user, group, or computer object in Active Directory. This command must be run as root when creating a computer object.

`vastool`

vastool options

create

-g

-c *container*

-p *password*

-i *info*

-x

name

`vastool create` will interpret the specified name, and then create different types of objects according to the format of the name. To create a computer object, the name must be formatted as "host/{hostname}". To create a computer object for the local computer and use it's hostname, just specify "host/". If the name does not start with "host/", the name will be interpreted as a user name if `-g` is not specified. The `-g` option will cause the name to be interpreted as a group name, and a group object will be created.

The new user, group, or computer object will be located in the default users or computer containers in Active Directory. You can create the new object anywhere in the Active Directory tree by using the `-c` option to specify the distinguished name (DN) of a container to create the object in.

When creating a user or a group, the new user or group will not be Unix enabled by default, unless you use the `-i` option. By specifying an information string with `-i`, you can specify the information for the user's/group's Unix account. This string should be formatted as an entry in /etc/password or /etc/group. When creating a user, you can specify that user's new password with the `-p` option. Unless the `-x` option was specified, the newly created user will also be forced to change their password during their first login.

`vastool create` must be run as root when creating a computer object for the computer that vastool is running on. Part of the computer creation process is setting the computer object's password so that vascd can authenticate to Active Directory. This key is stored in a secure file that can only be accessed by root at `/etc/opt/vas/host.keytab`. `vastool create` does not need to be run as root when creating users or groups.

Also, when creating a computer object without using just "host/" as the name, you will need to add an option to `/etc/opt/vas/vas.conf` to get the SCO Authentication client components to use this name as the name of the computer object. For example, if you wanted the name of your computer object to be "host/mycomputer", in the `[libdefaults]` section, you will need to add a line that looks like:

computer_name_override = host/mycomputer

Without this configuration file setting the SCO Authentication clients will not be able to authenticate to Active Directory correctly. This is done automatically for you when running `vastool join` with the `-n` option.

The user you authenticate to Active Directory as must have the appropriate administrative privileges in order to create the new user, group, or computer object. Computer object creation can be delegated to other users besides Administrators. To accomplish this, the Active Directory administrator must initially create the computer object in Active Directory using `vastool create` or `vastool join`. Then, the administrator can give another user rights to reset that computer object's password. This will allow that user to reinstall SCO Authentication without the administrator. In this situation, vastool will recognize that the computer object for this computer already exists, and will just try to reset the computer's password.

Following are two examples of user creation, two examples of group creation, and two examples of computer creation.

vastool create -c "OU=Engineering,DC=example,DC=com" jdoe

vastool create -i "jdoe:x:1001:1000:John Doe:/home/jdoe:/bin/bash" jdoe

vastool create -g marketing

vastool create -g -i "marketing:x:1005:john,mary" marketing

vastool -u admin create host/

vastool -u admin create -c "OU=Engineering,DC=example,DC=com" host/

vastool delete

`vastool delete` can be used to delete users, groups, computer objects, and NIS Map objects in Active Directory. This command must be run as root when deleting the computer object

`vastool`

vastool options

delete

-g

-n

-d

name...

You must have the appropriate administrative rights to delete objects in Active Directory. The names passed in will be interpreted according to the options used. The `-g` option will interpret the names as group names and the Active Directory objects for those groups will be deleted. `-n` will interpret the names as NIS Map objects. `-d` will interpret the names as LDAP Distinguished Names.

If no options are specified, then the names will be interpreted as user names, unless the names start with "host/", then the names will be interpreted as computer names. To delete the computer object for the computer `vastool delete` is running on, use "host/" as the computer name. You must have root access to delete the computer object for the local computer, since the key for the computer is removed from `/etc/opt/vas/host.keytab`.

Following is one example of group deletion, one example of user deletion, one example of NIS Map deletion, two examples of computer deletion, and an example of deleting an ldap object.

vastool delete -g eng

vastool delete jsmith

vastool delete -n hosts

vastool -u admin delete host/

vastool delete host/server.example.com

vastool delete -d "CN=Foo,DC=example,DC=com"

vastool flush

`vastool flush` can be used to clear the vascd cache. This command must be run as root.

`vastool`

vastool options

flush

-r

accounts | auth | keytab | users | groups

Flushing the accounts cache will remove all cached user, group and NIS Map information. This will force vascd to do complete lookups the next time it receives any NSS IPC requests. Flushing the auth cache will remove all cached user passwords. These are stored as SHA1 hashes in a secure file that is only accessible by root. Flushing the keytab will delete the SCO Authentication keytab file. Flushing the users cache will delete all cached user account information, and flushing the groups cache will delete all cached group information.

The users and groups cache will be regenerated after being flushed, unless the `-r` option is specified.

If you do not specify an argument to `vastool flush`, then the accounts and auth arguments will be implied, and all user/group account information, NIS Map information, and cached passwords will be deleted.

On systems that do not support PAM and NSS, then as part of flushing the users and groups, they will also be unmerged from the and files as well. If the caches are regenerated, then the users and groups will be merged back in.

vastool group

`vastool group` can be used to modify group membership lists.

`vastool`

vastool options

group

group_name

add | del

name...

You must have enough administrative privileges to modify the group object in Active Directory. Using the `add` option will add the listed users to the specified group. The `del` option will remove the specified users. Note that these changes will only appear on the Linux/UNIX systems if the group and users used in the command have been Unix-enabled. The changes will occur in Active Directory regardless of whether or not the users and groups have been Unix-enabled.

Please note that if the specified members do not already exist in Active Directory, then those names will not be added or removed from the group membership list.

Following is an example of adding the jsmith user to the eng group, and removing the jsmith user from the eng group.

vastool group eng add jsmith

vastool group eng del jsmith

vastool join

`vastool join` is a convenient command that wraps all of the necessary steps to configure the SCO Authentication client on a computer into one. It configures your realm, creates a computer object in Active Directory, and configures PAM and NSS. This command must be run as root.

`vastool`

vastool options

join

-c *container*

-n *computer_name*

-f

realm_name

servers...

`vastool join` will internally call `vastool configure realm` to configure the realm, `vastool create` to create a computer object in Active Directory for the computer, `vastool configure pam` to configure the PAM subsystem, and `vastool configure nss` to configure the NSS subsystem. The `vascd` client daemon will then be started. For more information on each of these steps, see their respective sections in this document.

The `-c` option will allow you to specify a container where your new computer object will be created. If that is not specified, then the computer object will be created in the default computers container. The `-n` option allows you to specify a different name for the computer object than what vastool would generate from your hostname. If `-n` is used, then a special parameter will be set in to denote to all the SCO Authentication components which name should be used for the computer object. This is the `host_principal_override` variable which should go in the `[libdefaults]` section. This parameter is written automatically for you. The computer name specified with the `-n` option should not be in the "host/{hostname}" form- it should just be a name which is an alternative to the system's hostname. It also must not contain any '.' characters- this name is not the FQDN.

Following the options, you must specify your Kerberos realm, which will be the same as your Active Directory domain. The services for this domain will be automatically detected through DNS and LDAP lookups. If you do not intend to use DNS, or it is not configured, you must specify a list of servers for your domain after the realm name.

If a computer object already exists in the directory for the computer name you are trying to use, an error will be reported. To override the existing computer object, use the `-f` option. In this case, the computer object's authentication key will be reset. Any other systems authenticating as that computer object will no longer be able to authenticate after the authentication key is reset.

Following is an example of `vastool join` using all of the defaults, then an example of joining a computer with a name other than it's hostname into a non default container

in an environment where DNS is not properly configured.

vastool -u admin join example.com server.example.com

vastool -u admin join -c "OU=Testlab,DC=example,DC=com" -n test_server example.com server.example.com

vastool kinit

`vastool kinit` can be used to obtain Kerberos tickets.

`vastool`

vastool options

kinit

service

If no arguments are specified, then the Kerberos TGT is obtained if it is not in the user's ticket cache. If services are specified, then those tickets will be obtained. If the `-u` vastool option was not specified, then these tickets will be stored in the user's ticket cache.

`vastool kinit` can be used to debug problems with Kerberos authentication. For example, to test if `vascd` can authenticate to Active Directory, you would run, as root,:

vastool -u host/ kinit

Using the vastool -s option, you can use the vastool kinit command as an authentication API from scripts and other programs which do not use PAM or the SCO Authentication API. This can be done by running:

vastool -u jdoe -s kinit

and then writing jdoe's password to stdin of the vastool process. The exit code of the vastool process will be 0 on a successful authentication, and 1 if authentication failed.

If you see any error messages, then vascd could not authenticate to Active Directory.

vastool klist

`vastool klist` can be used to list all of the tickets currently in the calling user's Kerberos ticket cache.

`vastool`

klist

The tickets in the user's ticket cache are printed to stdout. They will show the name of the service each ticket is for, the time the ticket was issued, and the time the ticket will expire. The ticket cache will be stored as a file owned by the user with permissions of 0600 at $HOME/.krb5cc or in /tmp/krb5cc_{user's uid}.

vastool kdestroy

`vastool kdestroy` will destroy all of the tickets that are in the calling user's Kerberos ticket cache.

`vastool`

vastool options

kdestroy

A user's Kerberos ticket cache is a file owned by the user with permissions of 0600 that will be either at $HOME/.krb5cc or in /tmp/krb5cc_{user's uid}. Normally, the user's Kerberos TGT is stored there along with any other tickets that have been obtained. These tickets can all be cleared with `vastool kdestroy`.

vastool list

`vastool list` can be used to list the users and groups that are stored in Active Directory.

`vastool`

vastool options

list

-a

-l

-f

-c

users | user {username} | groups | group {groupname}

By default, `vastool list` will not directly use LDAP, but will use vascd to lookup the group and user accounts. This allows `vastool list` to take advantage of vascd's information cache. The `-l` option will force `vastool list` to directly use LDAP and bypass the vascd cache.

The `-a` option will list all the users or groups in Active Directory, not just the Unix enabled ones. This will automatically also set the `-l` option and force `vastool list` to directly contact Active Directory over LDAP. When `-a` is used, only the name attributes will be listed. Not using `-a` will list all of the attributes for Unix enabled groups and users.

The `-f` option will force `vascd` to update it's cache immediately, without respecting the ldap blackout period. The `-c` will not send an update IPC to `vascd`, but just read directly from the cache. This may result in old data that is incorrect.

`vastool list user` and `vastool list group` will only list the user/group specified. The `-l` still applies and will force LDAP lookups. The `-a` will search all users and groups, not just the unix enabled ones.

Following are examples of listing the Unix enabled groups that `vascd` knows about, the Unix enabled users `vascd` knows about, and listing the Unix enabled users obtained directly from Active Directory.

vastool list groups

vastool list users

vastool list -l users

vastool load

`vastool load` can be used to import existing users and groups.

`vastool`

vastool options

load

-f *file*

-c *container*

-p *password*

-x

users | groups

`vastool load` will read from a file if the `-f` option is specified, otherwise it will read from stdin. The input must follow the format of if loading users. If loading groups the input must be formatted like /etc/group. You can load the users or groups into any Active Directory container using the `-c` option. Otherwise, they will be created in the default users container.

Please note that existing passwords cannot be imported into Active Directory. If `-p` is used to specify a password when loading users, all of the new users will have their passwords set to the specified password. Otherwise, a random password made up of alphanumeric characters will be generated for each user. These generated passwords will be stored in a file the administrator can use to notify the new user's what their password is. Unless the `-x` option was specified, the newly created users will be forced to change their password during their first login. Passwords cannot be set for groups, and the `-p` option is ignored when loading groups.

Any errors will be logged to stderr and a log file whose location will be printed out at the end of the `vastool load` operation. It is very important to ensure that the UID's and GID's specified for the users and groups being imported do not conflict with existing users and groups already in Active Directory. The import process does not check for conflicts before creating the new users and groups.

When importing groups that have members, those members need to be created first. For example, for the following group entry: `group1:x:1400:user1,user2,user3`, you should first import user1, user2, and user3. Otherwise, when the group object is created in Active Directory none of the members will be stored in the group. This is due to the fact that group membership lists in Active Directory are stored as lists of distinguished names, and those DN's cannot be looked up if the group members do not already exist.

The following is an example of importing a file of users into a specific Active Directory container, and setting all of their default passwords to "change.me".

vastool load -f /tmp/newusers.txt -p change.me -c "OU=eng,DC=example,DC=com" users

vastool merge

`vastool merge` can be used to merge Active Directory users and groups into the local and files. This should only be done on systems that do not support NSS and PAM. This command must be run as root.

`vastool`

vastool options

merge

users | user {username} | groups

`vastool merge users` will merge Active Directory user accounts into /etc/passwd. `vastool merge user` will merge only the given user into /etc/passwd. `vastool merge groups` will merge Active Directory groups into /etc/group. If no option is specified to `vastool merge`, then both Active Directory users and groups will be merged into and respectively.

`vastool merge` will ask `vascd` to update the vascd passwd cache or the vascd group cache, then merge the Active Directory users or groups into the existing passwd and group files. Merging will not occur if the vascd cache has not been updated since the last merge- this is determined by comparing the timestamps between the local account files and the vascd cache files. To force the merge to occur, use the `-f` option.

Account merging is not necessary for operating systems that support PAM and NSS. This functionality is provided to allow for account synchronization on platforms that do not allow you to use NSS and PAM. Also, account merging is done automatically by the SCO Authentication login utility when users attempt to login- in this case you will not need to manually run `vastool merge`.

It is not possible to synchronize users' passwords using `vastool merge`. Only their user account information will be stored in Due to this, you must use the applications bundled with the SCO Authentication client software to allow the Active Directory users to have access to the operating system where PAM is not supported.

Following is an example of merging the Active Directory user accounts, and an example of merging the Active Directory groups.

vastool merge users

vastool merge groups

vastool nis-import

`vastool nis-import` can be used to create NIS Map objects in Active Directory.

`vastool`

vastool options

nis-import

-o

-f *filename*

-c *container*

-s *format_string*

NIS Map Name

A NIS Map normally represents some standard configuration file such as `/etc/hosts`. `vastool nis-import` can read in a file specified with the `-f` option, or if

`-f` is not specified, then the NIS Map contents will be read from stdin. The `-c` option allows you to specify the Active Directory container to create the NIS Map object in. If `-c` is not specified, the NIS Map object will be created in the default computers container. The container that the NIS Map is created in is significant since only the computers in that same container will see the NIS Map. This behavior allows you to emulate the behavior of NIS domains by enabling you to distribute maps of the same name to different sets of computers.

The NIS Map Name argument is the name of the NIS Map object. This should normally be the name of the file you are importing. For example, if creating a NIS Map object for `/etc/hosts`, you would use "hosts" as the name.

The following is a list of the currently supported NIS Maps files with the map names that use those files:

`passwd`	passwd.byname, passwd.byuid
`group`	group.byname, group.byuid
`hosts`	hosts.byaddr, hosts.byname
`ethers`	ethers.byaddr, ethers.byname
`aliases`	mail.aliases
`protocols`	protocols.byname, protocols.bynumber
`services`	services.byname, services.byservice, service.byservicename
`rpc`	rpc.byname, rpc.bynumber
`netgroup`	only netgroup is supported

The `-s` option allows you to use a custom NIS Map that is not listed among the currently supported NIS Maps. You must specify a format string that will be stored with the NIS Map so that `vastool ypcat` knows how to parse the NIS Map. Each line in the NIS Map file that is not a comment or empty will be parsed as NIS Map entry. The

format string specifies how each line should be parsed to generate the NIS Map keys and the NIS Map values. The format string must be formatted as follows: "version:key index:separator char:key in value flag:comments at end flag" where:

version	is a version string denoting the format of the version. Currently only "v1" is supported.
key index	denotes the index in the string where the key value is. This should be a 0 based index, or it can follow the format of "index+" meaning that the key is the substring starting at index and taking up the rest of the string. This index is determined by columns decided by the separator character which follows. Note that this index is not the index of a normal array character, but the index of the columns that are determined by the separator character.
separator char	The separator char denotes how the lines in the file are split up into columns. It can be a regular character, '\0' to denote whitespace (any whitespace character), or '\1' to denote any non alpha-numeric character.
key in value flag	should be '1' if the key is part of the value string, and '0' if the value does not include the parts of the string which make up the key.
comments at end flag	should be '1' if each line can have a comment starting with '#' at it's end, and '0' otherwise. If the flag is '1', then if a line contains a '#' character, everything after it is ignored;

For example, the format string used for the passwd.byname NIS Map would look like:

"v1,0,:,1,0"

Please note that you can not import a NIS Map for passwd or group. Users and groups must be imported into Active Directory with `vastool load`. `vastool ypcat` will use the cached information for users and groups from `vascd` when ypcat'ing the passwd or group NIS Maps.

The following is an example of importing a NIS Map for the hosts file.

vastool nis-import -f /home/jsmith/nisMaps/hosts hosts

vastool passwd

`vastool passwd` can be used to change your password, or to reset another user's password if you have enough administrative privileges.

`vastool`

vastool options

passwd

-s

-c

user_name

On some platforms, such as United Linux based Linux distributions and Solaris 8/9, the system passwd change utility does not work correctly when the `vas` NSS module is listed in `/etc/nsswitch.conf`. SCO Authentication users will not be able to change their passwords using the system passwd command. `vastool passwd` can be used by SCO Authentication users as a workaround.

If no user name is specified, then the calling user's (or the user specified with the `-u` vastool option) password will be changed. If a user name is specified and `-s` is not specified, then `vastool passwd` will attempt to set the specified user's password- you must have enough administrative rights to modify that user's password. `vastool passwd` will change passwords in Active Directory using the Kerberos change password protocols. If a user name is not specified, the calling user's password will be changed. The user must supply their current password, and then provide a new password.

The `-s` option will allow you to use `vastool passwd` in a non-interactive mode. This allows you to use `vastool passwd` from web pages, scripts, and other tools that can obtain a user's current and new password, and change that user's password when that user does not have shell access. Since this option is intended to be used by another program on behalf of the user, the behavior of `vastool passwd` is changed.

You must specify the user's user name on the command line, and then write the user's current password, a new password, and the verification of the new password to stdin of the `vastool passwd` process. `vastool passwd` will then change the user's password.

The `-c` option requires a user name and also must be run as root. This will change the cached password for the given user. This is useful mainly for debugging purposes. It does NOT change the user's password in Active Directory, only in the local cache.

One important note is that when using the `pam_vas` PAM module with disconnected authentication enabled, then `vastool passwd` will not be able to sync up the user credential cache with the new password for the user. On systems where the passwd utility correctly works, the user's changed password will be synced up correctly in the user credential cache.

The following are examples of changing your own password, setting the jsmith user's password, and using the `-s` option to run `vastool passwd` from a script or web page for the jsmith user.

vastool passwd

vastool passwd jsmith

vastool passwd -s jsmith

vastool realms

`vastool realms` can be used to query the network for the Active Directory domains and also will detect the domain controllers on the network. This information can be stored in the realms cache- to do this you must be root.

`vastool`

vastool options

realms

find [root | services [realm] | gc] | cache {update | list}

`vastool realms find` will detect the all of the domains on your network. `vastool realms find root` will find the root domain. `vastool realms find services` will find all of the services for all of the domains, or one of the domains if specified. `vastool realms find gc` will find all of the global catalogs on the network.

`vastool realms cache list` will print out the information that is stored in the realms cache. This cache is used to decrease the amount of DNS traffic that must be performed. `vastool realms cache update` will detect all of the available information and refresh the realms cache.

vastool timesync

`vastool timesync` can be used to query the time on the Active Directory server and synchronized the system clock with the Active Directory server. In order to set the system clock this command must be run as root.

`vastool`

vastool options

timesync

[timeserver] [-q]

Running `vastool timesync` without specifying a timeserver will automatically use the Active Directory server that was configured using the `vastool join` or the `vastool configure realm`commands.

Use the `-q` option to query the server's time with out setting the system clock.

vastool unconfigure

`vastool unconfigure` can be used to remove the SCO Authentication configuration from the NSS and PAM subsystems. This command must be run as root.

`vastool`

vastool options

unconfigure

nss | pam [service...]

Running `vastool unconfigure nss` will remove the vas settings in `/etc/nsswitch.conf`. Running `vastool unconfigure pam` without specifying any service names will remove the SCO Authentication configuration from all PAM enabled services. If service names are specified, only those specified services will have their SCO Authentication configuration removed.

Running `vastool unconfigure` with no arguments will unconfigure both the NSS configuration and all PAM enabled services from using SCO Authentication.

The following are examples of removing the SCO Authentication configuration from `/etc/nsswitch.conf`, disabling SCO Authentication authentication from the ssh server, and disabling SCO Authentication authentication for all PAM enabled services.

vastool unconfigure nss

vastool unconfigure pam sshd

vastool unconfigure pam

vastool unjoin

`vastool unjoin` is a convenient command that wraps all of the steps to remove your computer object from Active Directory and to remove the SCO Authentication configuration from the NSS and PAM subsystems in one step. This command must be run as root.

`vastool`

vastool options

unjoin

-n *computer_name*

`vastool unjoin` will first remove the NSS and PAM configurations with `vas-`

`tool unconfigure`. It will then prompt you for your administrative password in order to delete the computer object for your machine in Active Directory. As part of this process, `vascd` will be stopped.

If you used the `-n` option in the `vastool join` process, then you need to specify that same name that you used in the join in the `vastool unjoin` process.

The following are examples of unjoining the machine where the computer object is named after the system hostname, and unjoining the machine when the computer object name does not match the hostname.

vastool -u admin unjoin

vastool -u admin unjoin -n computer_name

vastool unmerge

`vastool merge` can be used to remove Active Directory users and groups from /etc/passwd and /etc/group. This should only be done on systems that do not support NSS and PAM. This command must be run as root.

`vastool`

vastool options

unmerge

users | groups

`vastool unmerge users` will remove all Active Directory user accounts from /etc/passwd. `vastool unmerge groups` will remove all Active Directory groups from /etc/group. If no option is specified to `vastool unmerge`, then both the Active Directory users and groups will be removed.

Following is an example of removing Active Directory user accounts, and an example of removing the Active Directory groups.

vastool unmerge users

vastool unmerge groups

vastool ypcat

`vastool ypcat` can be used to view NIS Map objects that are stored in Active Directory. It does not use the NIS protocol, but uses the SCO Authentication client daemon, `vascd`, to access the NIS Map objects over Kerberos/LDAP. It is functionally equivalent to the standard ypcat.

`vastool`

vastool options

ypcat

-k

-x

NIS Map Name

The `-k` option will print out the NIS Map keys. Without `-k`, only the values are printed out. The `-x` option will print out the NIS Map nickname translation table that `vastool ypcat` uses. These NIS Map objects must be imported into Active Directory using `vastool nis-import`.

You do not need to authenticate to Active Directory when using `vastool ypcat`. `vastool ypcat` will send an IPC to vascd who will get the NIS Map contents. `vascd` will be able to use it's LDAP cache, making the ypcat process very efficient.

The following are examples of viewing the map nickname translation table, viewing the hosts NIS Map with it's keys.

vastool ypcat -x

vastool ypcat -k hosts

See Also

pam_vas (5), vascd (1)